Universidade Nova de Lisboa

Faculdade de Ciências e Tecnologia

Departamento de Informática

MediaAccess - Componentes para Acesso a Informação Multimédia na Web

Por

André Filipe Correia Rosa

Nº 29315

Dissertação apresentada na Faculdade de Ciências e Tecnologia
da Universidade Nova de Lisboa
para obtenção do grau de Mestre em Engenharia Informática

Orientador

Prof. Doutor Nuno Robalo Correia

Lisboa

2009

Agradecimentos

Em primeiro lugar, gostaria de agradecer aos meus pais, pois sem eles não seria possível elaborar esta dissertação. Graças às condições por eles proporcionadas, bem como os seus conselhos, mesmo não parecendo os mais acertados, consegui obter sucesso em diferentes ocasiões. Para além dos meus pais, gostaria de agradecer à minha irmã e ao resto da minha família, de sangue e do "coração", por todo o apoio que forneceram desde sempre.

Agradeço também, e em especial, à minha namorada por todas as palavras de apoio, motivação e noções de Design dadas durante a realização desta dissertação. Acredito que seja complicado partilhar a vida com alguém que vive num mundo à parte, onde tudo se baseia em linhas de código, mas parece que ela sabe qual é o segredo!

Para além das principais fontes de apoio acima mencionadas, gostaria também de agradecer a todas as pessoas que duvidaram de mim, dando-me a força extra que por vezes é necessário para ultrapassar os obstáculos ou desafios, como lhes prefiro chamar, que surgem no decorrer da vida.

Muito obrigado ao meu orientador, o professor Nuno Correia, por me dar a possibilidade de elaborar uma tese na área da Computação Multimédia, possibilitando-me o aprofundamento de conhecimentos em vários assuntos pelos quais tenho grande interesse, partilhando sempre o seu apoio e conhecimento. Obrigado ainda por me introduzir no Interactive Multimedia Group (IMG), onde me possibilitou lidar com excelentes pessoas quer a nível profissional quer a nível pessoal.

Por fim, mas com enorme apreço, gostaria de agradecer a todos os meus colegas que durante vários anos me acompanharam na Faculdade, formando um excelente grupo de trabalho e para além disso um bom núcleo de amigos. Eles sabem quem são, sem ter que mencionar nomes. Muito obrigado amigos!

Resumo

A Internet tem-se vindo a afirmar cada vez mais como um suporte para a partilha de conteúdos multimédia, havendo actualmente grande destaque para o vídeo. Quem tira grande partido destas características são os seus utilizadores, agrupando-se em redes sociais onde trocam experiências através de imagens, vídeos e áudio.

Actualmente já existem inúmeras páginas Web que permitem a partilha de diversos tipos de informação multimédia e são utilizadas por milhares de utilizadores a nível mundial. Porém, só com a melhoria das condições de acesso à Internet é possível desfrutar desses conteúdos com maior qualidade e rapidez, bem como adicionar novas funcionalidades para os utilizadores tirarem partido.

Esta dissertação de mestrado visa então propor a criação de ferramentas para a construção de arquivos audiovisuais para acesso e participação pública. Esta participação consiste na contribuição dos utilizadores com materiais audiovisuais de modo a integrar e aumentar o arquivo. O utilizador, para além de contribuir com materiais, poderá aceder ao conteúdo disponível a partir de uma interface criada com ferramentas independentes desse mesmo arquivo.

As ferramentas criadas serão baseadas em tecnologias Web de forma a serem reutilizáveis e multi-plataforma. Terão em consideração a utilização de conteúdos de diferentes domínios, como por exemplo vídeos e imagens retirados de diferentes contextos, para permitir que as componentes e ferramentas desenvolvidas possam ser testadas em várias situações e diferentes comunidades de utilizadores.

Abstract

Internet is the main support for multimedia content sharing, with recent emphasis on video. Currently, users are the ones taking advantage of these capabilities, participating in social networks where they share experiences through images, videos and audio.

There are several Web sites that allow sharing of multimedia content and are used globally. With the evolution of the Internet, now it is possible to enjoy all these contents faster than before, with better quality and new features that improve the user experience.

This master thesis proposes the development of tools to build multimedia archives for public access and participation. This participation consists in uploading content to the archive and accessing it, using an interface built with tools independent of the archive. Additionally, if the developed tools give the chance to edit the content of the archive, the user experience will be improved.

The tools will be developed based on Web technologies to allow reutilization and multiplatform access. To test the tools and components, archives from several domains will be used, including images and videos from multiple sources, providing a test framework in several environments and different user communities.

Índice de Matérias

Índice de Figuras

Índice de Tabelas

Introdução

Actualmente os documentos existentes no repositório global que é a Web, já não são apenas texto como no início, contêm os mais variados exemplos de multimédia, sendo possível encontrar páginas Web com grandes colecções de imagens, áudio e, cada vez mais, vídeos. Os criadores dessas páginas utilizam esses elementos não só para enriquecer o seu conteúdo, mas também para melhorar a sua apresentação.

A crescente utilização de vídeos e imagens com maiores resoluções e de melhor qualidade, deve-se ao facto dos acessos à Internet que existem hoje à disposição dos utilizadores terem uma maior largura de banda, permitindo um acesso mais rápido e capacidade de transferir mais conteúdo em menos tempo. Um exemplo desta evolução é a comparação das velocidades/taxas de transmissão das antigas ligações à Internet, as ligações *Dial-up*, com a mais recente tecnologia *Wireless-N* (IEEE 802.11n) onde as velocidades cresceram dos 56Kbits/s para os 300 Mbit/s.

Ao contrário do que acontecia inicialmente, os documentos partilhados já não têm um público-alvo tão específico devido à globalização do acesso à Internet. Qualquer pessoa pode visualizar uma página Web criada por alguém noutra parte do mundo, visto que o acesso à Internet e à Web fazem hoje parte do quotidiano de milhões de pessoas, sendo um fenómeno social.

Antes deste desenvolvimento acontecer já existiam arquivos multimédia, que consistem em bases de dados que armazenam quantidades elevadas de conteúdos multimédia e informação relacionada com os mesmos, fornecendo aos utilizadores a possibilidade de pesquisar e consultar os dados existentes no arquivo. Claro que o aparecimento das tecnologias, bem como as evoluções mencionadas anteriormente, também tiveram impacto nos arquivos multimédia, mais precisamente na forma como os utilizadores acedem a estes. As limitações físicas inerentes às

redes de computadores, foram ultrapassadas devido à Internet e o acesso a conteúdos remotos foi melhorado através da utilização de interfaces Web, tornando-se visualmente mais agradável.

As interfaces Web, ou as páginas disponibilizadas na Web, que se podem visitar hoje em dia, são aplicações que para além de uma apresentação apelativa, possibilitam que os utilizadores efectuem operações através de um simples *browser*, integrados com outros serviços de Internet como o email (ex.: aplicações da Google como o Gmail e o Google Docs). Ao serem executadas em *browsers* são suportadas por diferentes plataformas. No que diz respeito a este tipo de aplicações, os mais recentes avanços, e que representam uma motivação adicional, caminham para a criação de aplicações Web denominadas Rich Internet Applications (RIAs), que contêm propriedades e funcionalidades de aplicações de *desktop*, podendo ser executadas fora de um *browser.*

Descrição e Contexto

Como foi referido anteriormente, a Internet é utilizada de várias formas pela sociedade. Para além da sua utilização em ambientes de trabalho, a Internet é cada vez mais explorada pela sua vertente lúdica, em actividades como a conversação, os jogos online e a interacção social.

Um dos mais recentes serviços que a Internet permite utilizar são as redes sociais, que suportam as comunidades online, grupos de indivíduos que utilizam a Internet para partilhar os mesmos interesses ou actividades. A maior parte destas comunidades são baseadas em páginas Web, usadas para trocar mensagens instantâneas, *emails* e partilhar conteúdos multimédia, cativando a atenção dos utilizadores e fazendo com que o número de registos neste tipo de comunidades esteja continuamente a crescer.

Os utilizadores das comunidades online utilizam diversas formas para partilhar os seus conteúdos, quer sejam imagens, áudio, vídeo ou texto. No que diz respeito às imagens e ao texto, estes elementos normalmente são visualizados na página, havendo a possibilidade de descarregar posteriormente esse conteúdo para os computadores pessoais. Já para o áudio e o vídeo, existem duas possibilidades para disponibilizar esses conteúdos para os utilizadores, *download* ou *streaming*. O *download* não permite ao utilizador visualizar/ouvir os conteúdos imediatamente após o carregamento da página, pois só quando estes se encontrarem no computador pessoal é

que terão essa hipótese. O *streaming* dá a possibilidade de os utilizadores reproduzirem áudio e vídeo, sem que seja necessário esperar que o *download* termine completamente.

Actualmente existe um termo muito referido, mas que não tem uma definição aceite universalmente, Web 2.0. O termo apareceu pela primeira vez em 2004 numa conferência (Web 2.0 Conference, 5-7 Outubro de 2004, São Francisco) e desde então tem existido alguma agitação à volta deste. Apesar de ainda haver algum desacordo quanto ao termo, é possível dizer que os utilizadores e as comunidades online criadas por estes são centrais nesta nova definição da Web. Neste contexto, situam-se os objectivos desta tese, que passam pela construção de ferramentas baseadas em tecnologias Web para a partilha de conteúdos multimédia, que serão explicadas com mais detalhe na próxima secção.

Solução Apresentada

Tendo em conta o que foi anteriormente descrito, e depois de entender o problema totalmente, é imprescindível efectuar as pesquisas necessárias para estudar as tecnologias existentes, entender quais os benefícios e desvantagens de cada uma delas e fazer escolhas para que as tecnologias escolhidas optimizem ao máximo a solução a apresentar. Para além das tecnologias relevantes, é do interesse para a realização da dissertação tomar conhecimento dos trabalhos que se destacam nesta área.

Esta solução passa por criar um arquivo multimédia para acesso e participação pública, permitindo armazenar imagens e vídeos de diversas origens. O arquivo será de participação pública pois os utilizadores terão que contribuir com materiais audiovisuais, para que este possa aumentar e consigam tirar o máximo partido de toda a aplicação.

Para gerir toda a informação associada aos conteúdos do arquivo multimédia será necessário o desenvolvimento de uma base de dados, havendo a necessidade de obter mais informações sobre os Sistemas de Gestão de Base de Dados (SGBDs), bem como definir qual a melhor estrutura para a base de dados a ser implementada, de modo a que as pesquisas funcionem o melhor possível. É importante tomar conhecimento destes sistemas, para que as ferramentas possam criar componentes independentemente do SGBD que guarda a informação sobre os conteúdos.

Visto que as ferramentas a serem desenvolvidas poderão ser integradas numa aplicação que visa a interacção social, os utilizadores irão tirar partido da aplicação para efectuar algumas acções comuns nas páginas Web de comunidades online, como por exemplo efectuar comentários aos conteúdos. A utilização das componentes desenvolvidas, em conjunto com outras tecnologias permitirá a ligação entre a o arquivo multimédia e a interacção social, dando a possibilidade aos utilizadores de inserirem metadados, classificando os conteúdos existentes no arquivo e facilitando a pesquisa.

Para além das funcionalidades acima descritas, pretende-se também que os utilizadores das ferramentas desenvolvidas possam configurar as componentes não só no aspecto funcional (acesso aos dados guardados no arquivo), mas também o aspecto visual (posicionamento dos elementos constituintes de uma componente) quando esta opção se justifique.

Por fim, para que os utilizadores consigam interagir com as componentes desenvolvidas, é indispensável uma interface que sirva de intermediário entre o utilizador e as componentes. Esta interface permitirá testar as funcionalidades das componentes, quer individualmente quer em conjunto. Permite ainda testar o seu funcionamento em conjunto com outras tecnologias e obter *feedback* sobre a usabilidade da interface.

Principais Contribuições Previstas

As contribuições previstas com a realização desta tese são:

- Estudo do estado da arte e levantamento de tecnologias existentes – é necessário efectuar uma pesquisa onde seja conhecida a abordagem e perspectiva de outros grupos de investigação existentes a trabalhar em áreas semelhantes, bem como fazer um levantamento das tecnologias actuais que possam ser úteis para cumprir os objectivos desta tese;

- Definição de requisitos (a partir de exemplos) para este tipo de aplicações Web – depois de efectuar o estudo e levantamento mencionado acima, serão tiradas

conclusões de modo a entender melhor o que será necessário para desenvolver as componentes e como estas deverão interagir com utilizador;

- Desenvolvimento de componentes para a construção de aplicações Web – desenvolver as componentes necessárias para que os utilizadores possam desfrutar dos conteúdos existentes, permitindo ainda a edição/montagem de vídeos existentes no arquivo;

- Validação das ferramentas desenvolvidas através de um protótipo – Após o desenvolvimento de todas as componentes necessárias, efectua-se a integração num protótipo de aplicação, para que as componentes sejam testadas numa situação real.

Estrutura do Documento

Nesta introdução à dissertação foram mencionados aspectos como uma breve descrição e o contexto onde se insere esta dissertação. Foram também apresentadas propostas de solução, bem como os objectivos e principais contribuições previstas com a elaboração desta dissertação.

O restante documento está organizado em 5 capítulos, onde no capítulo 1 é apresentado um resumo do estado da arte, que por sua vez se encontra dividido em várias secções como Arquivos Digitais, Editores de Vídeo, Metadados e Tecnologias Relevantes. Após o capítulo 1, que também serve para contextualizar melhor este trabalho, o capítulo 2 apresenta o modelo das componentes desenvolvidas, sendo estas explicadas com maior detalhe no capítulo 3.

O capítulo 4 mostra a forma como as ferramentas e componentes podem ser integradas num contexto real, apresentando e explicando como as componentes utilizadas em conjunto com outras tecnologias, podem criar uma aplicação Web funcional.

No último capítulo são apresentadas as conclusões retiradas da elaboração desta dissertação bem como o trabalho futuro que poderá ser desenvolvido tendo este como base.

1. Trabalho Relacionado

Este capítulo apresenta outros trabalhos de investigação e tecnologias que servem de inspiração e poderão ser utilizadas nesta tese. Os principais temas abordados ao longo deste capítulo serão arquivos digitais, editores de vídeo, metadados e tecnologias relevantes para a elaboração desta tese.

1.1 Arquivos Digitais

Já há algum tempo que se fazem estudos e investigações sobre as potencialidades dos arquivos digitais, de modo a aumentar e melhorar a partilha de documentos e consequentemente aumentar o nível de conhecimento das comunidades que utilizam estes arquivos. Como foi dito anteriormente, a Internet afirmou-se como um meio de comunicação e fonte de acesso a informação, possibilitando que alguns arquivos existentes se interligassem, formando um catálogo de informação distribuída de dimensão global e útil a um maior número de pessoas interessadas.

É notória a utilidade que os arquivos digitais podem ter em campos como a educação, ciência ou até mesmo no entretenimento. Ao aliar estes arquivos com as potencialidades da Internet, torna-se possível disponibilizar a nível global conteúdos como vídeos de aulas, experiências científicas ou simplesmente um filme, contribuindo não só para a evolução do conhecimento e tecnologias, como também para o aumento do conhecimento global.

John R. Smith [1] concentrou-se em estudar arquivos de vídeos digitais, chegando à conclusão que estes têm que conseguir integrar vários aspectos, como a catalogação e a pesquisa de conteúdos, mantendo a eficiência necessária para servir o utilizador da melhor maneira possível. Na Figura 1.1, apresentam-se as diferentes fases de interacção com um arquivo, onde podem ser tomadas decisões que levantam algumas questões e desafios.

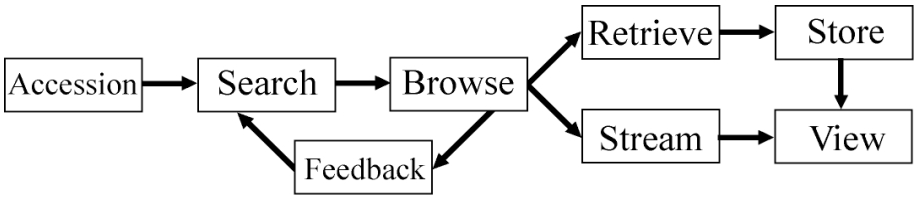

Figura 1.1 – Modelo de interacção entre utilizador e arquivo de vídeos digitais

Numa primeira fase, *Accession*, onde são adicionados os conteúdos ao arquivo, é necessário tomar decisões que irão afectar o restante processo de interacção. É necessário decidir sobre o modo como vão ser armazenados os dados, sendo essencial garantir que são guardadas informações sobre os próprios conteúdos, como por exemplo, a localização do vídeo e o tipo de compressão utilizada. Esta informação poderá ser utilizada mais tarde nas pesquisas efectuadas pelo utilizador.

Nas fases seguintes, *Search* e *Browse*, o utilizador interage directamente com o arquivo, procurando os conteúdos desejados. Aqui torna-se evidente a importância da escolha do tipo de informação complementar a ser guardada. Um exemplo desta informação poderá ser a sumarização de vídeos, fornecendo as principais imagens para que o utilizador tenha uma ideia geral do conteúdo ou os metadados associados aos dados (como uma breve descrição do conteúdo, o(s) autor(es) ou palavras-chave que os identifiquem), permitindo que estes possam ser categorizados.

Após as fases mencionadas anteriormente, as restantes estão relacionadas com a forma como o utilizador acede aos conteúdos. No caso exemplificado na Figura 1.1, referente a vídeos, a escolha passa por decidir se o utilizador faz o *download* do vídeo, fase de *Retrieve*, armazenando-o localmente ou se o visualiza através de um processo de *streaming*, fase de *Stream*, em que o processo de reprodução de vídeo é efectuado através da Internet.

Um dos mais antigos e significativos exemplos de arquivos digitais, foi desenvolvido pela Universidade de Carnegie Mellon e é conhecido pelo nome de Informedia [2]. Na realidade o projecto Informedia tem uma dimensão enorme, sendo que a parte que diz respeito aos arquivos de vídeos digitais pode ser encontrada mais concretamente no Informedia Digital Video Library, que se divide em Informedia-I Digital Video Library (Informedia-I) e Informedia-II Vídeo Digital Library (Informedia-II). Na Figura 1.2 é possível observar o projecto Informedia de um ponto de vista global, sendo possível também avaliar a dimensão do projecto.

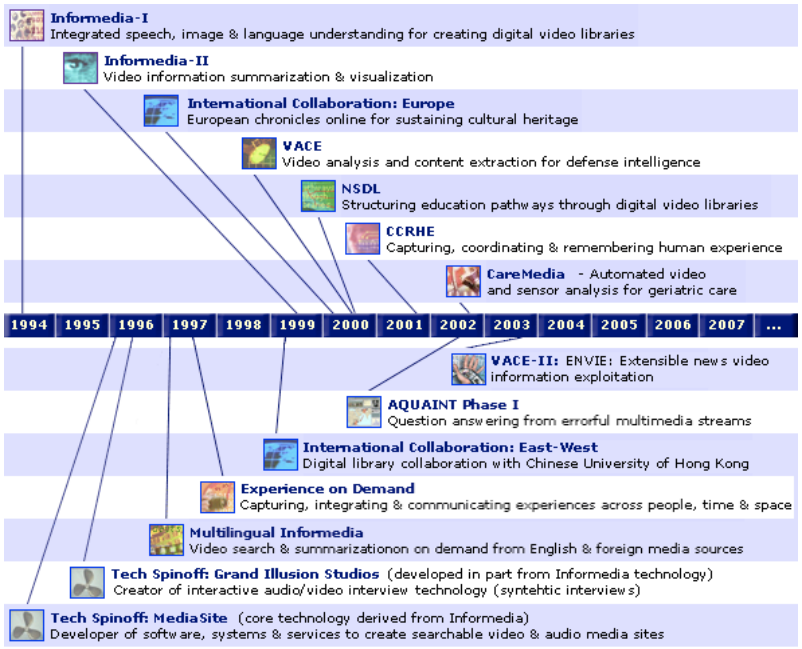

Figura 1.2 - Visão global do Projecto Informedia

No que diz respeito ao Informedia-I, este foi criado para desenvolver novas tecnologias de pesquisas em grandes colecções de dados, seleccionando os resultados mais relevantes de acordo com a pesquisa efectuada de forma a melhorar e tornar mais independente o acesso e exploração de informação por parte dos utilizadores.

Inicialmente, e tal como é referenciado em [2], o arquivo tinha um tamanho de aproximadamente 3 Gb de informação, entre os quais alguns milhares de horas de vídeo digital. Foram desenvolvidos processos automáticos para permitir a pesquisa através de conteúdo e garantir a fiabilidade dos resultados num arquivo de grandes dimensões. Os processos automáticos tiram partido de métodos de reconhecimento de voz/linguagem e processamento de imagem, para que seja possível transcrever, segmentar e catalogar os vídeos originais. Após aplicar os métodos de processamento de áudio no vídeo, passa a haver uma representação textual da parte audível do vídeo, tornando mais fácil obter os vídeos como resultados de uma pesquisa textual, através da correspondência de palavras transcritas do áudio com a consulta efectuada.

Ao utilizar esta técnica de reconhecimento de voz/linguagem e processamento de imagens, o Informedia-I introduziu o conceito de *video skimming*. Este processo consiste em fazer um resumo do vídeo (entre os 5 e os 20% da duração original do vídeo), permitindo um

rápido visionamento deste através da selecção das imagens e componentes áudio mais importantes (por exemplo, por objectos específicos ou palavras-chave inseridas no áudio).

No Informedia-II os responsáveis pelo projecto decidiram continuar com os principais objectivos do Informedia-I, nomeadamente no que diz respeito à transcrição de áudio dos vídeos, mas ao mesmo tempo também quiseram melhorar os métodos que analisam as imagens dos vídeos, aperfeiçoando os resultado das pesquisas. Apesar da semelhança entre ambas as versões do Informedia, foram introduzidas algumas novidades nas tecnologias criando novos modos de visualização dos resultados, tal como se pode verificar nas Figura 1.3, 1.4 e 1.5.

Figura 1.3 - Interface apresentando resultados após uma pesquisa no sistema do projecto Informedia-I

Figura 1.4 - Interface com resultados de uma pesquisa com o texto "El Niño" no sistema do projecto Informedia-II

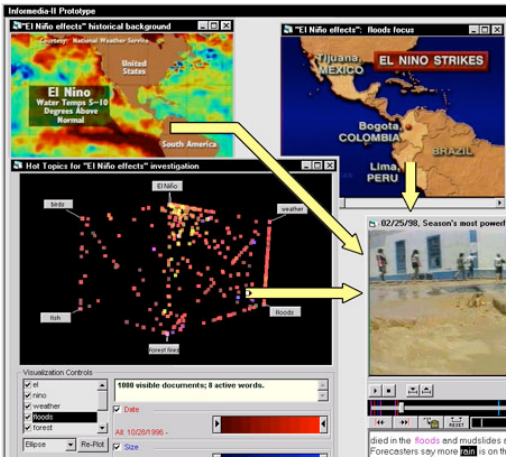

Figura 1.5 - Interface com novos modos de visualização apresentado os resultados à pesquisa "El Niño effects"

O Informedia-II, tal como referido em [3], trouxe um novidade que foi o facto de, através de métodos de reconhecimento de voz, ser possível interpretar e reconhecer nomes, localidades, datas e referências temporais, permitindo representações dos resultados como a apresentada na Figura 1.5, onde é possível explorar os resultados obtidos através das referências geográficas encontradas durante o processamento do vídeo. Para além desta novidade, a extracção de dados feita através do processamento do vídeo foi melhorada, ficando mais rápida e mais eficiente, dando uma maior relevância às técnicas de resumo dos vídeos.

Outro exemplo, mais recente, de um arquivo digital é o BilVideo que segundo Dönderler *et al.* em [4, 5], é o arquivo de vídeos digitais mais completo no que diz respeito a propriedades exploradas por este sistema. Para desenvolver o BilVideo, os seus criadores basearam-se numa arquitectura cliente/servidor, apresentada na Figura 1.6, onde o utilizador comunica com o servidor através de um *applet* Java.

Neste projecto não existe apenas uma base de dados com toda a informação, mas várias que repartem a informação entre si. Na base de dados *Raw video database* é guardada a informação do sistema referente aos vídeos. Em *Feature database* mantém-se a informação relacionada com os vídeos como, por exemplo, as palavras-chave associadas e na base de dados *Knowledge base* estão alojados metadados referentes à semântica e aos factos existentes no vídeo.

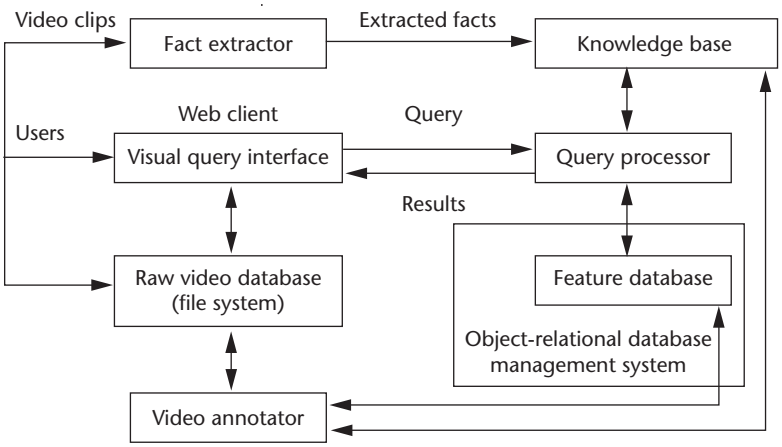

Figura 1.6 - Arquitectura do sistema BilVideo

A extracção de factos do vídeo é feita de forma semi-automática, utilizando uma ferramenta criada propositadamente para este projecto, que permite ao utilizador ver o vídeo e especificar os objectos nas imagens, através de rectângulos denominados *minimum bounding rectangles* (MBRs), como mostrado na Figura 1.7. No exemplo demonstrado, o utilizador está a definir dois objectos numa imagem do vídeo através de rectângulos, um para o veículo e outro para a bandeira no topo do mesmo. Esses objectos são depois caracterizados pela sua posição na imagem e por um nome.

É assim possível processar automaticamente as relações espácio-temporais bidimensionais dos objectos no vídeo, ao permitir que os utilizadores seleccionem os mesmos objectos nas imagens adjacentes. Estas relações dão ainda a possibilidade de introduzir a noção de movimento dos objectos. Para além deste tipo de informação, também é possível fazer anotações, ou seja, dados que identificam o vídeo, como por exemplo o título, a duração e o ano de produção.

As relações espácio-temporais mencionadas anteriormente constituem a maior inovação deste projecto, permitindo que as pesquisas sejam feitas de uma maneira muito diferente do que normalmente é apresentado aos utilizadores.

Figura 1.7- Interface de extracção de factos

A interface Web desenvolvida para os utilizadores do BilVideo permite efectuar dois tipos de pesquisas, a primeira baseada em condições espaciais e a segunda através da definição de trajectórias dos objectos. A pesquisa através de condições espaciais é feita através da especificação de MBRs, onde o utilizador indica o posicionamento pretendido dos objectos numa imagem do vídeo, sendo posteriormente comparado com as definições extraídas através do processo descrito no parágrafo anterior, como se pode verificar na Figura 1.8. Já nas pesquisas de trajectórias, o utilizador indica uma trajectória através de uma sequência de pontos, que podem ser modificados, apagados ou inseridos conforme o utilizador necessite. Os pontos inseridos servem para descrever o melhor possível a trajectória pretendida, podendo ou não ser exactamente igual à existente na base de dados, pois para além de definir a trajectória é dada a possibilidade de definir o valor de semelhança desejado. Este valor vai de 0 e 100% (50% por omissão), sendo possível observar a definição de uma trajectória para pesquisa na Figura 1.9.

Após o utilizador efectuar as pesquisas através das ferramentas visuais, são traduzidas para uma linguagem baseada em SQL que permite procurar os resultados no servidor, sendo esta também editável através da interface, como se verifica na Figura 1.10. Segundo os autores, as pesquisas através das ferramentas gráficas são direccionadas para os novos utilizadores do BilVideo, para que estes tomem conhecimento com a linguagem de pesquisa textual utilizada, passando a utilizar esta posteriormente.

Figura 1.8 - Pesquisa através de especificação espacial

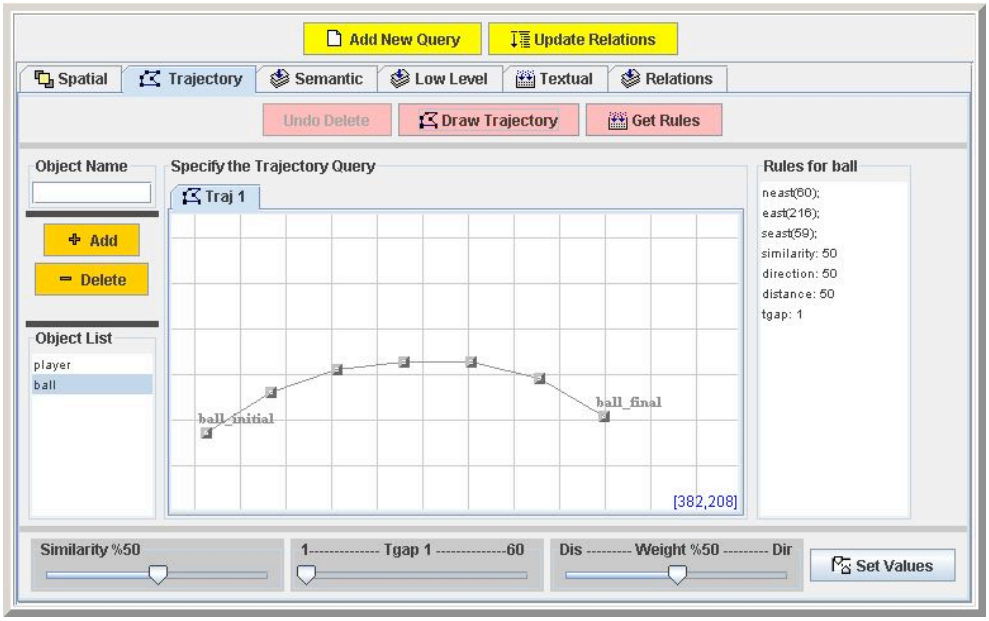

Figura 1.9 - Pesquisa através de especificação de trajectórias

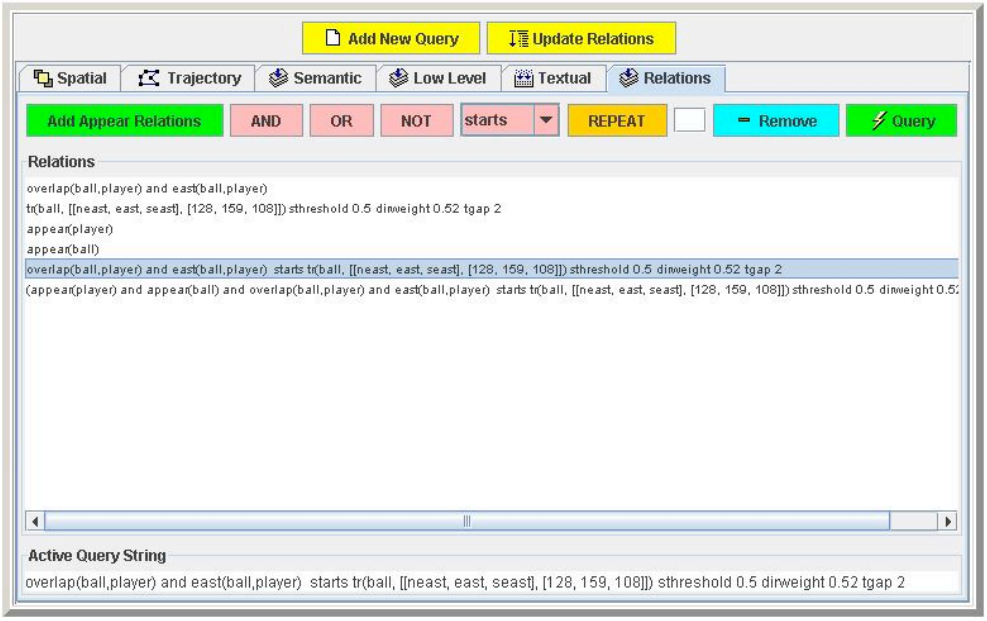

Figura 1.10 - Resultado final da pesquisa definida numa linguagem baseada em SQL

1.2 Editores de Vídeo

Se no passado ao falar de edição de vídeo se pensava em edição linear, corte e colagem de fitas, hoje em dia pensa-se em sistemas de edição não linear, processos computacionais e *software* de edição de vídeo. Actualmente existem vários exemplos deste tipo de programas, de livre utilização ou *software* proprietário, sendo um dos mais utilizados o Adobe Premiere/Adobe Premiere Pro, cuja interface é apresentada na Figura 1.11.

O Adobe Premiere Pro dá várias possibilidades aos seus utilizadores, tendo em conta que suporta a edição dos principais formatos de vídeo digital, como por exemplo *Digital Video* (**DV**), High Definition Vídeo (HDV), RED e Sony XDCAM, bem como vídeos Standard Definition (SD) ou High Definition (HD) obtidos através de placas de captura de vídeo, ambos numa resolução máxima de 4000x4000 *pixels*, 32 bits por cada canal de cor (RGB e YUV). Também é possível efectuar a edição de áudio com suporte para áudio 2.0 e 5.1 *surround sound* para um melhor resultado no que diz respeito ao áudio.

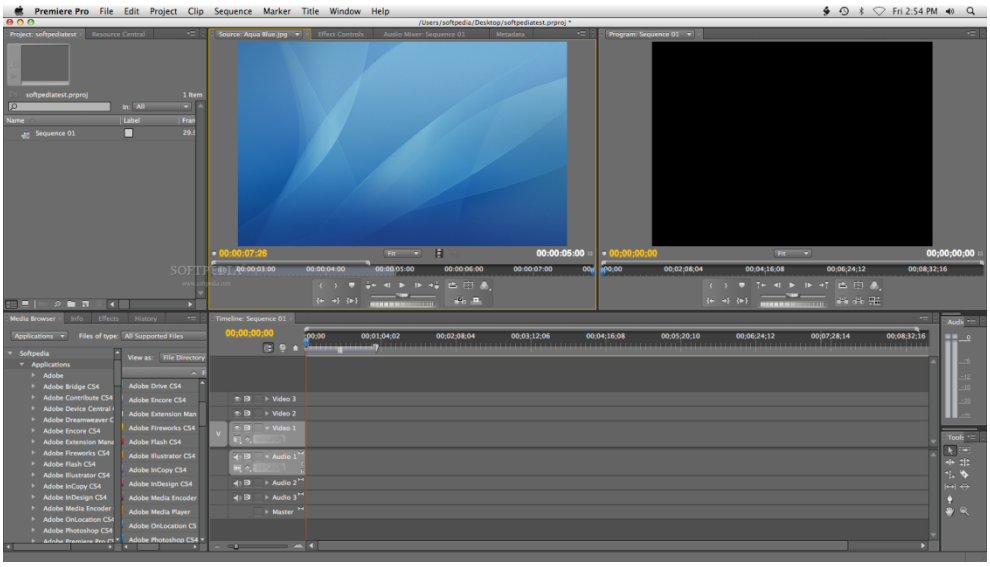

Figura 1.11 – Interface do *software* Adobe Premiere Pro

A última versão deste *software*, lançada no final de 2008, introduziu algumas novidades relativamente às anteriores. No que diz respeito aos formatos de *output*, a última versão permite gravar a edição num formato de alta definição, Blue-Ray, ou gravar logo uma versão Web, exportando para o formato do Adobe Flash (swf). No entanto a maior novidade que a Adobe apresentou é a transcrição do áudio dos vídeos para texto, para que posteriormente se possam pesquisar partes do vídeo através de texto.

Apesar de todas as propriedades e funcionalidades, este produto da Adobe tem um público-alvo bem demarcado, essencialmente composto por utilizadores com conhecimentos avançados ou profissionais da área do vídeo. Para o utilizador comum, que deseja editar os seus vídeos caseiros, mas não possui conhecimentos muito avançados poderá recorrer a editores como o Windows Movie Maker/Windows Live Movie Maker (Windows), iMovie (Mac OS) ou Kino (Linux). Estes editores têm, em comparação com o Adobe Premiere Pro, uma interface mais simples, como se pode verificar nas Figuras 1.12, 1.13 e 1.14, dando no entanto várias possibilidades de edição mas com algumas limitações (por exemplo o número de vídeos ou faixas de áudio editáveis em simultâneo e os formatos de importação e exportação suportados).

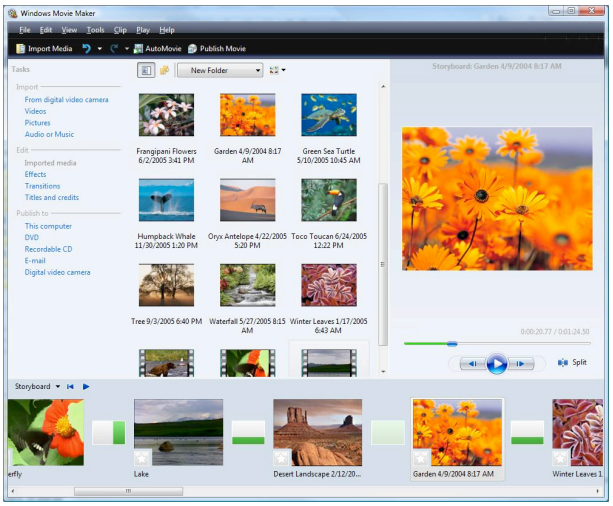

Figura 1.12 - Interface do Windows Movie Maker

Figura 1.13 - Interface do iMovie

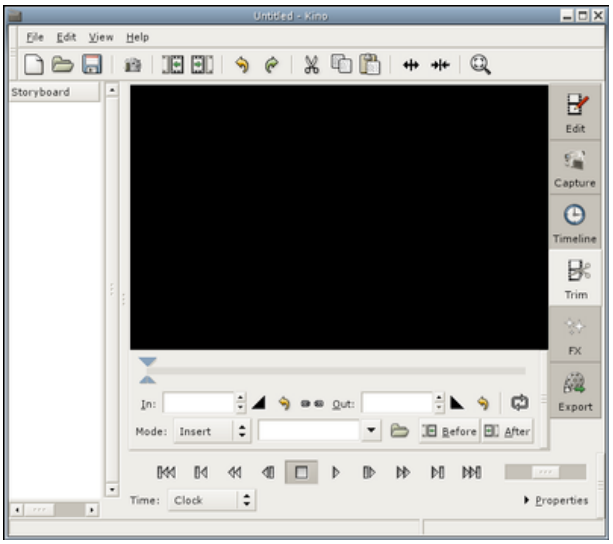

Figura 1.14 – Interface do Kino

Em 2001, J. Casares e Myers *et al.*[6,7], apresentaram um editor de vídeo que tira partido do arquivo digital Informedia, mencionado no subcapítulo anterior, utilizando os vídeos existentes neste arquivo. Os responsáveis pelo SILVER pretenderam com a sua criação permitir diferentes tipos de produção, no entanto o objectivo principal do SILVER é facilitar a elaboração de relatórios multimédia dos alunos do ensino secundário, efectuando pesquisas sobre os temas desejados no Informedia e obter resultados gráficos (vídeos e imagens) ou textuais (artigos de revistas ou jornais).

Para além da edição a partir dos conteúdos do arquivo, esta aplicação também permite a criação de conteúdos originais de duas formas diferentes. A primeira consiste na possibilidade dos utilizadores filmarem os seus próprios vídeos e no final editarem-nos de modo a produzir o resultado final. Na segunda, o utilizador poderá criar um guião através de um conjunto de imagens sequenciais (*storyboard*) e posteriormente gravar os vídeos para corresponder ao guião.

A interface existente permite visualizar os resultados das pesquisas e fazer a edição pretendida. Esta interface dispõe de vários modos de visualização para facilitar a tarefa dos utilizadores, como exemplificado na Figura 1.15. No entanto, ao olhar para esta figura com os vários modos activos, torna-se difícil de processar tanta informação à primeira vista, apesar de cada um destes ter a sua finalidade, de acordo com o que será explicado de seguida:

- **Search Results View –** Apresenta os vídeos existentes no Informedia, resultantes da pesquisa textual efectuada pelo utilizador;

- **Source and Project Views –** Fornece uma visão global do projecto, permitindo que o utilizador arraste os vídeos, do arquivo ou existentes no disco rígido, para adicioná-los ao projecto;
- **Transcript View –** Como já foi mencionado neste relatório, a componente áudio dos vídeos existentes no Informedia, contém uma representação textual, efectuada pelo sistema. Este modo de visualização permite ler essa representação textual;
- **Timeline View –** Neste modo, a informação dos vídeos é representada de uma forma interessante, em que os utilizadores para seleccionar partes de um vídeo dispõem de uma representação hierárquica de três níveis. No nível superior temos a representação do vídeo na sua totalidade e nos dois seguintes a representação das partes seleccionadas no nível acima;
- **Preview View –** Mostra uma pré-visualização da parte seleccionada pelo utilizador no modo Timeline Preview;
- **Subject View –** Este modo de visualização tem como objectivo dar a possibilidade ao utilizador de agrupar os vídeos segundo temas criados por ele.
- **Outline View –** Tal como no modo anterior, o utilizador organiza os vídeos de acordo com os temas criados por ele, mas desta vez utilizando uma árvore hierárquica parecida à que é utilizada no Explorador do Windows.
- **Storyboard View –** Aqui o utilizador elabora um guião para o vídeo original que pretende criar.

Para além da edição manual por parte dos utilizadores, o SILVER dá a possibilidade de realizar uma edição automática, permitindo efectuar uma selecção inteligente quando o utilizador selecciona o texto no modo Transcript View. A aplicação selecciona a parte do vídeo correspondente a esse texto e caso o utilizador o deseje, faz as junções dos diferentes vídeos de forma automática, tentando ajustar da melhor forma os conteúdos gráficos e áudio.

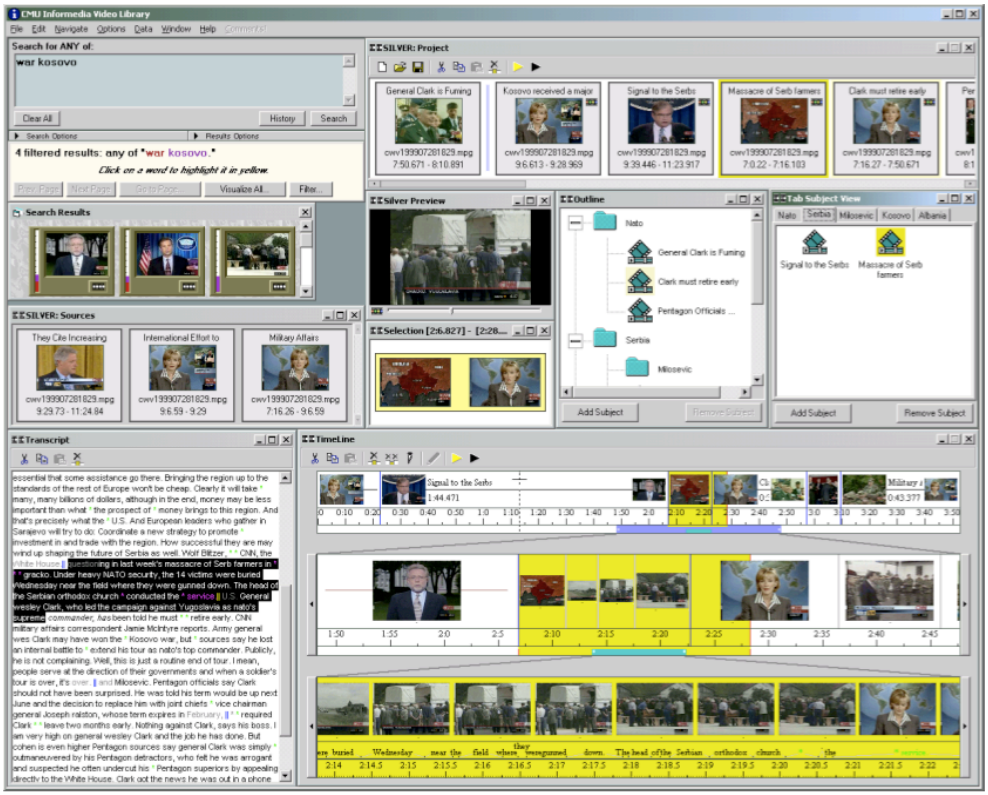

Figura 1.15 – Interface da aplicação SILVER com vários modos de visualização

O aparecimento deste tipo de aplicações que permite efectuar a edição não é novidade (a primeira versão do Adobe Premiere foi lançada em 1991). O que representa uma novidade no que diz respeito a edição de vídeo, tem a ver com o aparecimento de aplicações Web que permitem efectuar edição a partir de um *browser*. Recentemente, no princípio de 2007, a Adobe apresentou o Adobe Premiere Express. Esta aplicação Web tem como plataforma o Adobe Flash e foi desenvolvido em Adobe Flex (ver subcapítulo 1.4.2), permitindo aos utilizadores das páginas Web, que investiram nesta aplicação (*software* proprietário), editar vídeos apenas dispondo uma ligação à Internet e um *browser*. As funcionalidades do Adobe Premiere Express passam por criar um novo vídeo através da junção de outros vídeos, partes destes ou imagens, adicionando posteriormente títulos, molduras, transições entre os diferentes elementos (por exemplo *fade* ou *blur*) e um elemento áudio para utilizar como banda sonora. Na Figura 1.16, pode-se ver um exemplo da página Web do Photobucket [8], onde está integrado o Adobe Premiere Express.

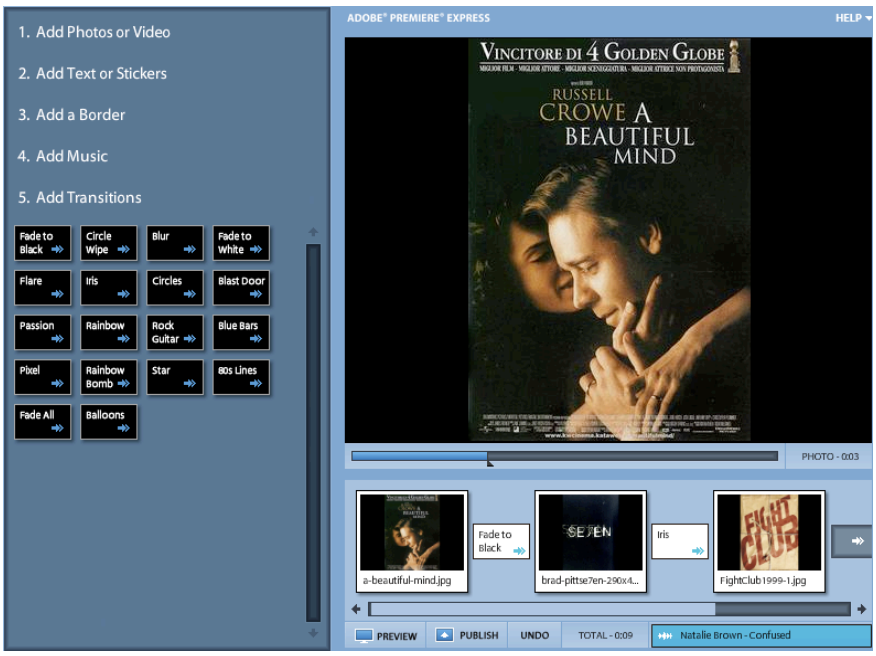

Figura 1.16 – Exemplo do Adobe Premiere Express integrado na página

1.3 Metadados

Os metadados são informação sobre a informação, ou dados sobre dados, em que o objectivo é caracterizar e contextualizar a informação ou dados a que estão associados. Este tipo de dados são utilizados para descrever informação de maneira a que esta possa ser identificada como apropriada para uma dada função. Podemos usar como exemplo o número 29315 que, sem conhecimento de outra qualquer informação associada ao mesmo, poderá corresponder a um peso, uma medida ou um valor monetário, mas se adicionarmos um dado descritivo como "número de aluno da FCT", então passamos a saber que na realidade este número identifica univocamente um aluno da FCT.

Os metadados podem ser utilizados em diferentes situações e aplicações. Actualmente os motores de busca na Web recorrem aos metadados, para melhorar a eficiência e a rapidez com que são apresentados os resultados. O laboratório de investigação da Yahoo! (Yahoo! Research) [9] e Baeza-Yates *et al.* [10], estudaram a relação existente entre o conteúdo das páginas e os metadados inseridos pelos criadores destas. Também apresentam a possibilidade de utilizar as

acções pesquisa/clique no link pretendido para gerar metadados que poderão ser analisados e posteriormente utilizados, por exemplo, em publicidade inteligente feita na Web.

Para além da Web, os metadados também são utilizados com outro tipo de finalidades. É possível encontrar metadados em ficheiros multimédia como fotografias digitais, áudio e vídeo. Nas fotografias digitais é possível encontrar o Exchangeable Image File Format (Exif) que permite utilizar os formatos JPEG e TIFF, para guardar informações sobre as fotografias tal como a marca da máquina com que foi retirada a fotografia, a resolução, a abertura do diafragma ou o tempo de exposição. A Figura 1.17 mostra um exemplo da apresentação desta informação.

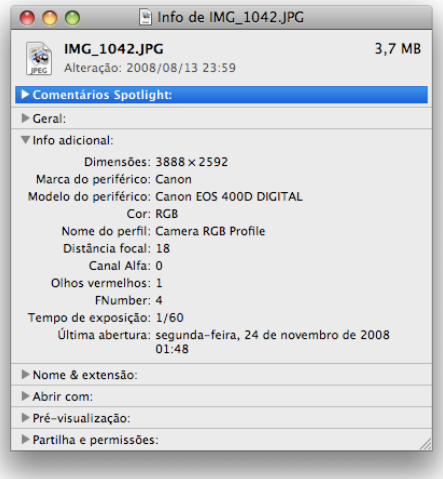

Figura 1.17 – Apresentação de metadados de uma fotografia digital

O formato de áudio digital MP3 também permite que sejam guardados metadados ao utilizar a norma ID3. Esta norma utiliza uma pequena parte da estrutura do ficheiro (128 bytes no ID3v1.1 e 227 bytes no ID3v1.2) e uma vez que é utilizada maioritariamente com ficheiros de música, permite guardar informações como o título da música, o artista, o álbum ou género da música. Mais uma vez os sistemas operativos podem aceder a essa informação, como exemplificado na Figura 1.18.

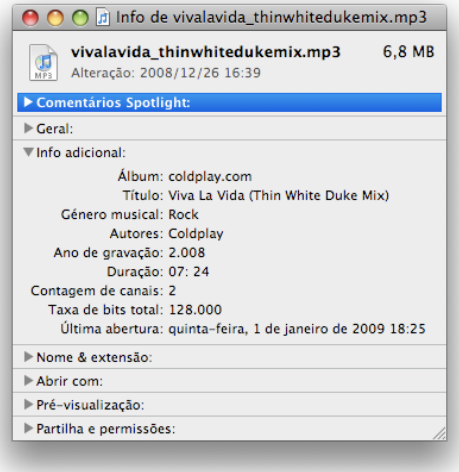

Figura 1.18 – Apresentação de metadados de um ficheiro áudio

No que diz respeito ao vídeo, existe uma norma criada pelo Moving Picture Experts Group (**MPEG**), o MPEG-7, para complementar as já existentes normas MPEG-1 MPEG-2 e MPEG-4. De acordo com [11], os objectivos desta norma passam por criar um conjunto de ferramentas que permitam descrever o conteúdo dos dados multimédia, para facilitar a tarefa a processos computacionais de análise de dados ou conversão destes, por exemplo converter de vídeo para texto, e criar uma estrutura para representar a informação multimédia, melhorando a sua transmissão, o seu armazenamento e a maneira como é encontrada pelos utilizadores. A maneira de representar o conteúdo difere das outras normas de metadados porque suporta diferentes níveis de abstracção, como por exemplo a textura dos objectos do vídeo, a sua cor, tamanho ou movimento/trajectória.

Para atingir os objectivos propostos, o MPEG-7 tem como base os seguintes elementos:

- **Descriptor:** representação de uma característica dos dados multimédia;
- **Description Scheme:** Estrutura e semântica das relações entre componentes, Descriptors ou Schemas;
- **Description Definition Language:** Linguagem necessária para a criação de novos elementos do ponto anterior, bem como a criação de Descriptors. O grupo responsável pelo desenvolvimento desta norma optou pela utilização do XML (mais detalhe no próximo subcapítulo) e criar extensões de forma a se adaptarem às necessidades.

- **System Tools:** As ferramentas criadas para o MPEG-7 servem para suportar as descrições, a sua sincronização com o conteúdo ou gerir e proteger a propriedade intelectual das descrições utilizadas na norma.

O MPEG também criou uma infraestrutura para conteúdos multimédia, MPEG-21 [12], centrado num objecto digital estruturado, identificado e com os respectivos metadados. O utilizador desta infraestrutura tem um papel importante, pois ele tanto pode ser criador de um objecto, consumidor ou distribuidor. O objecto digital utilizado, denominado Digital Item, na prática é uma combinação de estruturas, recursos e metadados. Os recursos deste objecto são os conteúdos multimédia existentes, os metadados para descrever ou contextualizar os conteúdos do Digital Item e uma estrutura que especifica as relações existentes entre os recursos e os metadados.

Na Figura 1.19 é apresentado um exemplo de um Digital Item, que consiste numa apresentação de uma universidade com fotografias, vídeos, texto, notícias de investigações realizadas e material de *e-learning*.

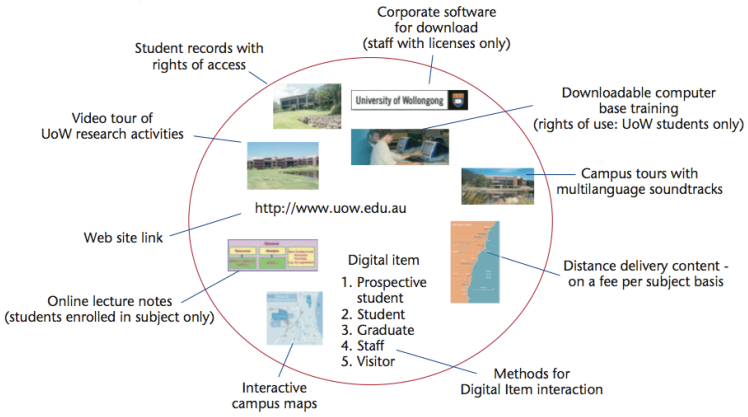

Figura 1.19 – Exemplo de um Digital Item

B. L. Tseng *et al.* em [13] utilizam ambas as normas mencionadas acima para construir um sistema de sumarização de vídeos, baseado numa arquitectura repartida em servidor, *midleware* e cliente.

Para além de uma versão adaptada do Digital Item existente na norma MPEG-21, os autores deste sistema utilizam as descrições existentes nas normas MPEG-7 e MPEG-21. Foram

criadas ferramentas para os utilizadores anotarem os conteúdos, permitindo adicionar etiquetas aos vídeos ou a partes destes e armazenar essa informação no servidor. Para utilizar estas ferramentas, os utilizadores interagem através de uma aplicação executada num PDA ou num *browser*, que permite efectuar pesquisas sob a forma de texto, tópicos ou restrições temporais.

No entanto, o cliente não comunica directamente com o servidor, utiliza a camada intermédia que contém dois subsistemas para efectuar a pesquisa e devolver os resultados ao cliente. O primeiro, denominado *personalization engine*, faz a correspondência entre a pesquisa efectuada pelo utilizador, com os metadados definidos no MPEG-7 e as preferências criadas com a adaptação do Digital Item do MPEG-21, para seleccionar os segmentos dos vídeos que irá devolver. O segundo subsistema, o *adaptation engine*, faz as adaptações necessárias aos resultados obtidos, de modo a que o utilizador possa visualizá-los independentemente de onde esteja a ser executado o cliente desta aplicação.

Até aqui, neste subcapítulo falamos de metadados como informação que caracteriza/contextualiza informação, dados ou conteúdos, necessitando que esses dados sejam introduzidos por um utilizador. Recentemente, têm surgido esforços para investigar e desenvolver processos que consigam fazer a extracção automática de características, que depois possam ser utilizadas para facilitar as pesquisas dentro de um arquivo de recursos multimédia.

Um exemplo destes processos pode ser verificado em [14], onde os autores propõem uma alternativa às anotações manuais por parte dos utilizadores, permitindo a pesquisa de uma colecção. Para tal, o sistema retira as características de baixo nível (*low level features*), como a textura e a cores dominantes das imagens. Para além da extracção destas características, o sistema foi previamente preparado com conceitos úteis para a pesquisa (por exemplo praia, interior, exterior, praia e neve), através da utilização de uma técnica chamada Regularized Least Squares Classifier, que permite fazer a correspondência entre os conceitos e as imagens, através de classificação binária resultante de funções matemáticas. Após os dois passos anteriores, o sistema permite ao utilizador efectuar pesquisas no conjunto de imagens, utilizando os conceitos pré-definidos e utilizados na preparação do sistema.

1.4 Tecnologias Relevantes

Esta secção irá apresentar algumas tecnologias e/ou aplicações que de alguma maneira são relevantes, ou estão relacionadas com a elaboração desta tese. Inicialmente será feita uma breve apresentação sobre Sistemas de Gestão de Base de Dados (visto ser necessário para o arquivo multimédia), passando depois às tecnologias e/ou aplicações úteis para a construção das componentes para aceder e interagir com o arquivo.

1.4.1 Sistemas de Gestão de de Base de Dados

Os Sistemas de Gestão de Base de Dados (SGBD) são aplicações que, como o nome indica, são utilizadas para organizar e gerir a utilização de uma base de dados, mais especificamente armazenar, actualizar ou consultar os dados existentes. A utilização destas aplicações torna a interacção com a base de dados mais fácil e interactivo, recorrendo muitas vezes a ferramentas gráficas.

O MySQL é um exemplo de um SGBD pertencente à Sun Microsystems, cuja base de utilização é livre, existindo algumas ferramentas ou versões com mais funcionalidades que têm de ser pagas. Esta aplicação foi desenvolvida em C e C++, é multi-plataforma e utiliza a linguagem SQL. Para interagir com a aplicação utilizando o SQL, o MySQL vem com uma linha de comandos e uma interface Web chamada phpMyAdmin, integradas no pacote base de instalação, podendo depois ser feito o download do pacote MySQL GUI Tools, que inclui interfaces gráficas como o MySQL Administrator e o MySQL Query Browser.

Este sistema é bastante utilizado em aplicações Web, devido ao facto de ser possível aceder às bases de dados através da linguagem PHP, embora estejam disponíveis bibliotecas para outros ambientes, como por exemplo Java ou .Net. Para além destas características, pode-se destacar outras como a reduzida exigência de recursos de hardware, suporte à utilização de *triggers*, *cursors*, *procedures*, *functions*, *views* com possibilidade de actualização, cache de consultas e sub-SELECTS (*nested* SELECTS).

Para além deste SGBD, brevemente descrito anteriormente, existem outros como o PostgresSQL, que comparando com o MySQL, também é de utilização livre e dispõe algumas ferramentas para melhorar a sua utilização. Uma das maiores diferenças entre eles consiste na forma como são guardados os dados no sistema onde está instalada a aplicação, pois o PostgreSQL dispõe apenas uma estrutura para guardar os dados (*storage engine*), enquanto que o MySQL contém nove estruturas diferentes para esse efeito. No entanto, ambos os sistemas são largamente utilizados quando as aplicações requerem a utilização de uma base de dados, devido à base de utilização livre, eficácia e desempenho demonstrados por ambos.

1.4.2 Adobe Flash Platform

Esta plataforma da Adobe [15] é composta por um conjunto de tecnologias, suportada por uma variedade de programas e comunidades de utilizadores. Juntas, estas tecnologias tentam fornecer tudo o que é necessário para desenvolver aplicações Web completas e inovadoras, capazes de executar num vasto leque de *browsers*, sistemas operativos e dispositivos. A Figura 1.20 apresenta uma visão global da relação entre esta plataforma e as aplicações Web.

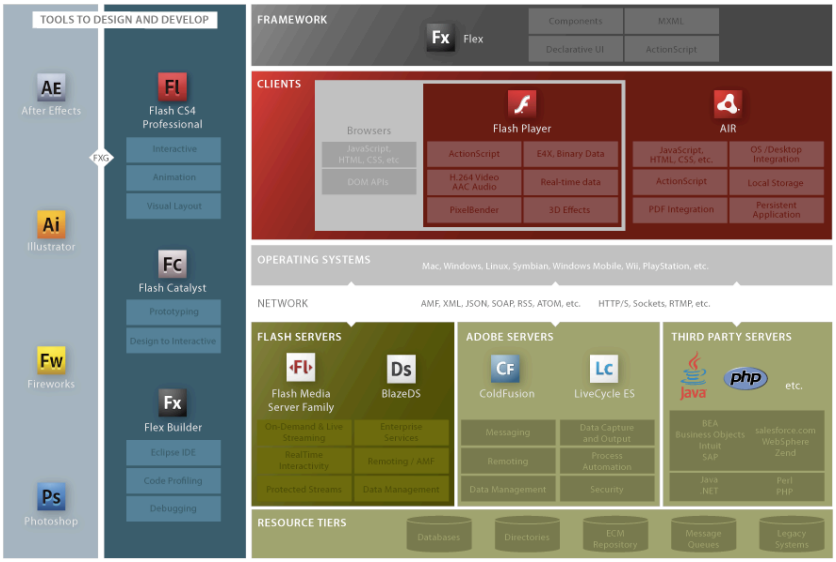

Figura 1.20 – Relação entre a plataforma Adobe Flash e as aplicações Web

Esta plataforma tem como base o Adobe Flash, anteriormente Macromedia Flash, que é uma tecnologia utilizada para criar animações, integrar conteúdos multimédia nas páginas Web (suporta *streaming* de áudio e vídeo) e mais recentemente para desenvolver Aplicações Ricas para a Internet (RIAs, do inglês Rich Internet Applications). Para executar as aplicações desenvolvidas com o Flash é necessário ter instalado o Adobe Flash Player, *standalone* ou *plugin* para *browser*.

Com o Flash é possível aliar a parte do design, através da criação ou importação de imagens e objectos vectoriais, com o desenvolvimento mais avançado e com mais recursos através de uma linguagem de programação chamada ActionScript. Esta linguagem é orientada a objectos e tem uma sintaxe semelhante à do Javascript, sendo utilizada para criar a maior parte da interactividade que pode ser encontrada numa aplicação de Flash.

A Adobe, para além do Flash disponibiliza o Flex, outra tecnologia que permite a criação de RIAs. Embora seja semelhante ao Flash, o Flex apareceu para minimizar os problemas de adaptação dos programadores tradicionais à experiência de animação e design do Flash, criando um ambiente de desenvolvimento mais familiar através da linguagem de programação MXML, baseada no XML e dando a possibilidade de integrar Actionscript no desenvolvimento. No entanto o Software Development Kit (SDK) do Flex já inclui algumas componentes de interfaces como botões, listas, caixas, utilização de *Web services* e suporte para *drag and drops*.

As aplicações criadas com o Flex tanto podem ser executadas no browser, tirando partido do Flash Player como num ambiente local utilizando o Adober AIR (ver adiante), que permite executar as aplicações localmente, aumentado o desempenho *online* ou *offline*. Um exemplo da utilização desta tecnologia é o, já mencionado, Adobe Premiere Express.

O Adobe Integrated Runtime, ou Adobe AIR como é mais conhecido, dá a possibilidade de executar aplicações Web num ambiente local, semelhante às aplicações normalmente instaladas pelos utilizadores nos seus computadores pessoais. Na Tabela 1, retirada de [16], é apresentada uma tabela de comparação entre a utilização de RIAs num *browser* e num ambiente local através do Adobe AIR.

FEATURE	RIAS IN THE BROWSER	RIAS ON THE DESKTOP
Application delivery	Applications can be easily discovered, explored, and used.	Installed applications have more persistence, power, and functionality.
Installation	No application installation is necessary.	Applications install seamlessly from the browser or download and install like a traditional desktop application.
Application updates	Applications are updated by pushing new content to a website.	AIR provides APIs that allow applications to be updated as easily as pushing new content to a website.
Multiple operating system support	Applications run on multiple operating systems and browsers.	AIR applications are cross-platform, so they can be installed on and run on multiple operating systems.
Programming languages	JavaScript is provided by browsers and ActionScript™ is provided by Adobe Flash® Player.	Integrated JavaScript and ActionScript virtual machines are compatible with the browser.
Background capability	RIAs can run only in a visible browser window.	Applications can run in the background or provide notifications like traditional desktop applications.
Persistence	Activity is limited to the browser session. When the browser is closed, information is lost.	RIAs are installed and available on the desktop. They store information locally and operate offline.
Desktop integration	Applications are sandboxed, so desktop integration is limited.	Applications can access a desktop file system, clipboard, drag and drop events, system tray/notifications, and more.
User interface control	RIAs run within a browser window that has its own controls, branding, and integration with the desktop.	RIAs have a customizable user interface and desktop integration, enabling branded experiences.
Data storage	Applications have limited local storage, which the browser can destroy.	Applications have unlimited local storage and access to a local database, plus encrypted local storage.

Tabela 1 – Comparação entre a utilização de *browsers* e ambientes locais para executar RIAs

1.4.3 Javascript

O Javascript é uma linguagem de programação, utilizada maioritariamente no desenvolvimento de páginas Web, que tira partido de ser executada no *browser*, ou seja, do lado do cliente, poupando recursos do lado do servidor.

É uma linguagem estruturada, rica e híbrida, que para além de suportar o paradigma imperativo, suportando elementos como cláusulas condicionais (*if* e *switch*) ou ciclos (*while* e *for*), também suporta o paradigma funcional e orientado a objectos, pois permite definir funções onde estas podem ser consideradas objectos, ou até construir funções que retornem funções.

Nesta linguagem, as variáveis declaradas não estão associadas a um tipo, pois a tipificação no Javascript é dinâmica. O tipo da variável depende do valor que a variável contém, pois inicialmente pode conter um número inteiro e depois passar a conter uma *string*, sendo possível fazer a verificação do tipo associado num dado momento.

Apesar de largamente utilizado, o Javascript também contém algumas vulnerabilidades e falhas de segurança que podem ser exploradas por *hackers*, permitindo alguns ataques como por exemplo a introdução de código nas páginas desprotegidas, executando-o posteriormente. Para além disto, quando o desenvolvimento vai além de programas mais triviais, fazer o *debug* torna-se complicado, uma vez que existem algumas incompatibilidades entre os *browsers*, em especial

no Document Object Model (DOM), sendo necessária a utilização de um *debugger* para cada *browser*. Actualmente já existem *debuggers* para os *browsers* mais comuns entre os utilizadores.

1.4.4 Extensible Markup Language (XML)

O XML [17] é uma meta-linguagem, utilizada em documentos que contêm informação estruturada, desenvolvida com o objectivo principal de permitir a partilha de dados através de vários sistemas, particularmente sistemas ligados através da Internet, e ser utilizado em diferentes aplicações, entre as quais as aplicações Web. Os documentos XML para além de serem de fácil processamento e interpretação em termos computacionais, devem ser perceptíveis por parte dos humanos, de modo a facilitar a sua criação. Com esta linguagem também se pretende separar a semântica da apresentação, pois as anotações servem para definir a estrutura semântica do documento, mas não especificam o modo com a informação é apresentada.

A definição da estrutura dos documentos XML pode ser feita através de Document Type Definitions (**DTDs**) ou através de XML Schema, que permite uma estruturação mais complexa do que os DTDs. Para a definição do modo como a informação é apresentada recorre-se a guias de estilo como Cascading Style Sheets (**CSS**) ou Extensible StyleSheet Language (**XSL**)/ Extensible Stylesheet Language Transformations (**XSLT**). Na Figura 1.21 é possível verificar a separação da estrutura, conteúdo e características de apresentação para gerar o documento final.

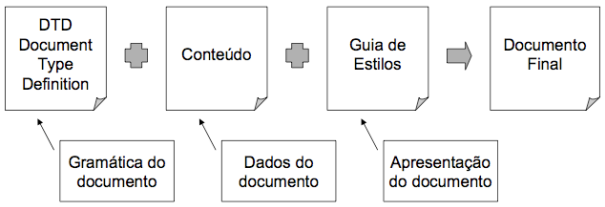

Figura 1.21 – Exemplo da separação de conceitos feita com a utilização de XML

Os documentos XML também permitem efectuar pesquisas ao seu conteúdo, através da utilização da linguagem XPath que se baseia num modelo de representação em forma de árvore do documento, seleccionando os nós pretendidos através de alguns critérios e relativamente a um nó de contexto. Além do XPath, existe a linguagem XQuery que também permite navegar nos

documentos XML, permitindo obter e manipular informação dos documentos XML, utilizando expressões XPath para especificar partes do documentos e obtê-las como retorno.

1.4.5 Asynchronous JavaScript and XML (AJAX)

O AJAX [18] foi criado para desenvolver aplicações Web ou RIAs e consiste num conjunto de ferramentas já existentes, mencionadas anteriormente, como o HTML e CSS para a apresentação, o Javascript e DOM para acederem aos dados inseridos nas páginas Web ou conteúdos de um documento XML alojado no servidor. Ao utilizar o AJAX é possível obter informação do servidor assincronamente (através do objecto XMLHttpRequest), onde assincronamente neste caso significa que comunicará com o servidor quando este se encontrar disponível, sem alterar o conteúdo da página que está a ser visualizada pelo utilizador.

Para além das ferramentas abrangidas pelo AJAX, o autor das aplicações pode recorrer, por exemplo, ao PHP do lado do servidor para gerar conteúdos dinâmicos ou XML e XSLT para processar os dados obtidos do servidor. Com isto é possível desenvolver aplicações Web mais interactivas, respondendo com maior rapidez aos pedidos do utilizador. Algumas partes das páginas Web podem ser recarregadas individualmente, transmitindo ao utilizador uma sensação de maior rapidez, mesmo sem alterar nada do lado do servidor. Na Figura 1.22, é apresentada uma comparação entre o modelo de aplicações Web tradicionais e o modelo de aplicações utilizando o AJAX.

Esta tecnologia ganhou projecção quando a Google passou a utilizar o XMLHttpRequest na suas aplicações Web Gmail e Google Maps, uma vez que a eficiência e impacto junto dos utilizadores foi notória.

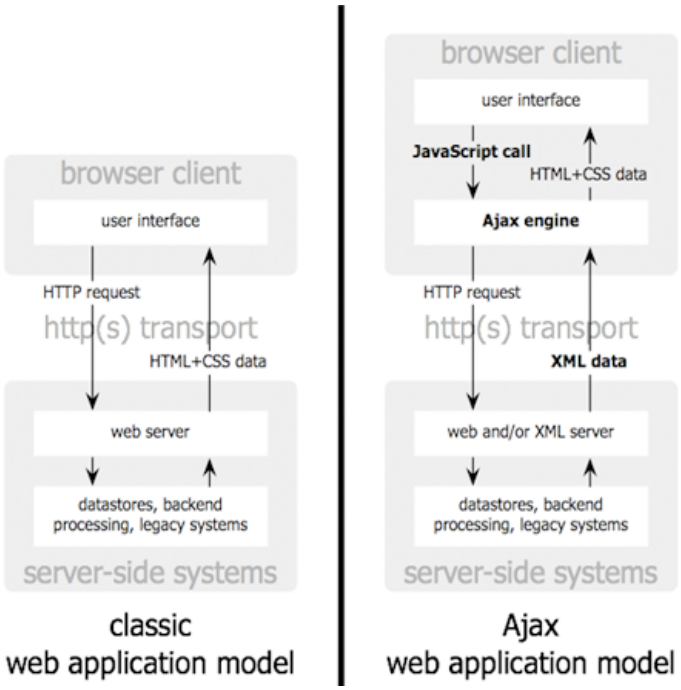

Figura 1.22 - Comparação entre o modelo de aplicações Web tradicionais e o modelo de aplicações utilizando o AJAX

1.5 Exemplos de Arquivos e Aplicações Web num Contexto Real

Nesta última secção do Trabalho Relacionado serão apresentados exemplos relacionados com esta tese, onde são utilizadas algumas tecnologias mencionadas anteriormente.

Um dos exemplos mais evidentes hoje em dia é o YouTube [19, 20], que consiste num arquivo com um enorme número de vídeos armazenados, que em 2006 já contava com 100 milhões de visionamentos por dia.

Para fornecer o serviço, o YouTube integra na sua arquitectura elementos como um Netscaler, que gere o tráfego do servidor e faz *caching* de algum conteúdo estático, de modo a acelerar todo o processo de carregamento por parte dos utilizadores. Ligados ao Netscaler estão os servidores Web que contêm o Apache, que por sua vez também se serve do Netscaler para ajudar a lidar com grandes quantidades de informação, com o mod_fastcgi activo, que suporta o protocolo FastCGI. Isto permite que os pedidos sejam passados do Apache para um servidor aplicacional desenvolvido em Python, que comunica com as bases de dados obtendo a informação necessária e formatando a página *html* que irá ser apresentada ao utilizador.

Estes servidores Web executam em ambiente Linux, mais propriamente SuSe, que contém um compilador de Python → C (psyco) para acelerar os processos desenvolvidos em Python. Mas nem todos os processos foram desenvolvidos em Python, visto que os processos mais exigentes (por exemplo cifrar os dados) foram desenvolvidos em C. No que diz respeito ao conteúdo, os vídeos, em vez de ser utilizado o Apache, os criadores do YouTube utilizam o lighttpd, com as configurações alteradas de modo a aceitar mais conexões, diminuindo o tempo de carregamento dos vídeos.

Por fim, no que diz respeito ao armazenamento dos vídeos, o YouTube começou a utilizar o MySQL para gerir a informação como os utilizadores, metadados e descrições, que funcionou correctamente até ao site começar a ser reconhecido e receber milhões de visitas por dia. Após isso, os responsáveis pelo YouTube tiveram que repensar a sua estratégia, passando a optimizar o sistema com técnicas, como por exemplo a optimização de consultas e utilização de *memcache*, guardando numa tabela de dispersão o resultado das consultas e diminuindo os acessos à base de dados ao verificar se o resultado já se encontra em *cache*. Para além destas optimizações, passaram a utilizar a replicação de bases de dados, distribuindo a carga que existia numa base de dados única por várias. Essa repartição não foi feita pelo conteúdo, mas sim pelos utilizadores, onde cada partição contém os dados referentes a um certo número de utilizadores e os vídeos que estes disponibilizaram.

No que diz respeito à interface Web utilizada, o YouTube tira partido do Adobe Flash Player para reproduzir os vídeos e apresentar algumas animações, sendo também possível verificar a utilização do AJAX, para modificar algumas partes da página rapidamente sem ter que a recarregar e sem fazer alterações no lado do servidor.

Existem outras aplicações que funcionam com o mesmo objectivo do YouTube, a partilha de vídeos, mas utilizam tecnologias diferentes no que diz respeito ao armazenamento dos dados no arquivo e a forma como estes são difundidos. Um exemplo disto é o Joost [21, 22], para partilhar vídeos de TV. Actualmente o Joost apenas dispõe de uma interface Web que permite a visualização dos vídeos, tal como acontece no YouTube, mas até Outubro de 2008, data da apresentação da interface Web para difundir vídeos, o Joost consistia numa aplicação que executava localmente no computador de cada utilizador. Para tal, essa aplicação contactava com os servidores do Joost, que em vez de devolverem os dados do vídeo, devolviam os endereços de outros utilizadores que já continham o vídeo. É neste pormenor que reside a diferença, pois esta

aplicação era baseada em protocolos de Peer to Peer (P2P), em que os utilizadores partilhavam os dados dos vídeos entre si, transferindo a carga dos servidores para o lado do cliente.

Outra interface Web que utiliza as tecnologias aqui mencionadas, tirando partido das potencialidades que estas oferecem é a página Web da MTV [23], que exemplifica perfeitamente a disponibilização de conteúdos multimédia a partir da Web, visto que nesta página os utilizadores podem aceder a vídeos (*videoclips* musicais), imagens (*wallpapers* ou fotografias) e texto (notícias do mundo da música) que fazem parte do arquivo da MTV. Esta página tira partido da plataforma Adobe, em particular o Adobe Flash Player, para reproduzir vídeos ou animações desenvolvidas em Flash. Uma vez que o público-alvo desta página Web é o mesmo do canal de Televisão MTV, que é um público mais jovem, a página Web chama a atenção ao utilizar as animações desenvolvidas em Flash e as mudanças de conteúdo dinâmicas através do AJAX.

2. Modelo de Componentes

No contexto desta tese, pode-se definir uma componente como um objecto que possibilita a interacção entre os utilizadores e um arquivo multimédia. As componentes desenvolvidas têm como objectivo permitir a inserção e acesso a conteúdos multimédia.

Para tal, as componentes e programas necessários para aceder ao arquivo foram desenvolvidas recorrendo a diferentes linguagens de programação e ferramentas utilizadas para criar aplicações Web, entre as quais se encontram o Flash, o Flex, PHP e o ActionScript.

As componentes, de acordo com o que se pretende, estarão a ser executadas do lado do cliente, ao passo que os conteúdos estarão alojados num servidor. Desta forma é possível definir a arquitectura baseada num modelo de cliente/servidor.

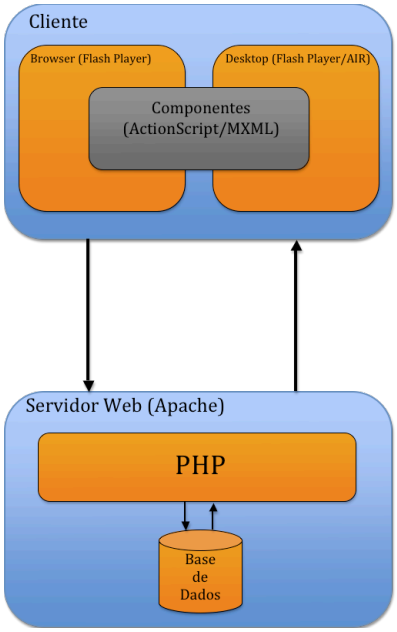

Figura 2.1 - Modelo cliente/servidor em que se baseia o sistema

Na Figura 2.1 é possível verificar que as componentes se encontram do lado cliente. Quando o utilizador acede a um componente através de um *browser*, recorre à utilização do Flash Player, versão *plugin* para *browsers*, em que a componente é descarregada para o computador do utilizador (para uma pasta temporária ou *cache*), poupando recursos do lado do servidor. Da mesma forma se o utilizador recorrer ao Flash Player, versão *standalone*, ou ao Adobe AIR para executar algumas componentes também está a poupar recursos do lado do servidor, pois as componentes já se encontram no seu computador.

As diferenças entre o AIR e o Flash Player podem ser vistas na Tabela 1, mas uma delas consiste num aspecto de segurança. Em ambos os casos podemos verificar a existência de uma *application sandbox*, que conta com diferentes definições para cada um destes ambientes de execução. No caso do Adobe AIR e visto que o principal objectivo é realizar aplicações *desktop*, a pessoa responsável pelo desenvolvimento da aplicação consegue aceder ao sistema de ficheiros do computador onde esta será executada, através de uma API desenvolvida para esse efeito. No capítulo 3.3 será explicada com mais detalhe uma das componentes desenvolvidas, que consiste numa aplicação *desktop* utilizada para inserir conteúdos no arquivo multimédia, bem como mais detalhes sobre o ambiente de execução AIR, no qual é executada esta componente.

Quer o utilizador execute as componentes através de um *browser* ou no *desktop*, para que este possa aceder aos conteúdos multimédia existentes no arquivo, será sempre necessário existir comunicação com um servidor. No servidor físico, é necessário que esteja a ser executado um servidor Web, por exemplo o Apache, para que alguns programas desenvolvidos em PHP funcionem correctamente, permitindo o bom funcionamento de todas as componentes desenvolvidas.

Como se pode observar na Figura 2.1, os programas realizados em PHP comunicam com uma base de dados que pode estar no mesmo servidor físico que o servidor Web ou noutro que esteja configurado para aceitar as comunicações. Apesar de se ficar com a ideia, depois de ver a Figura 2.1, que a base de dados se encontra no mesmo servidor, recorrendo à configuração inicial é possível definir o endereço da base de dados que se pretende utilizar.

Antes de se começar a utilizar as componentes é necessário que seja efectuada uma configuração inicial, feita através da componente destinada para tal. Na Figura 2.2 pode-se ver esta componente a ser executada na versão *standalone* do Flash Player.

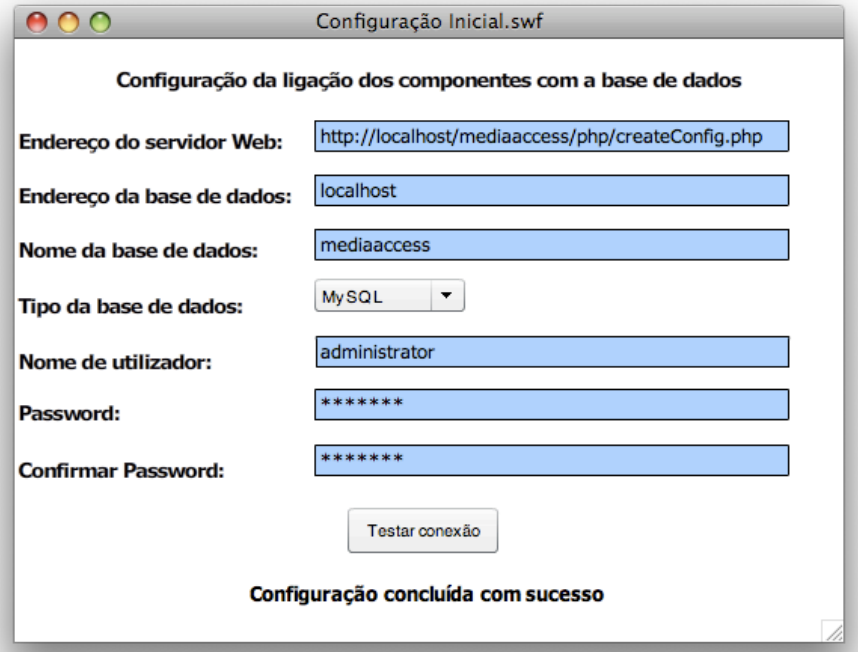

Figura 2.2 - Componente responsável pela configuração inicial

Na figura anterior é possível verificar o tipo de parâmetros necessários para concluir a configuração inicial. Esta configuração quando efectuada correctamente, permite o armazenamento das configurações que irão ser feitas através das ferramentas e o acesso aos conteúdos existentes no arquivo.

Ao executar esta configuração, sempre que os restantes ficheiros desenvolvidos em PHP necessitem de comunicar com a base de dados que suporta o arquivo multimédia, podem incluir o ficheiro gerado automaticamente por este configurador. Para além dos servidores Web e da base de dados é dada a possibilidade de escolher se a base de dados foi criada através de MySQL ou PostgreSQL. O código apresentado nas Figuras 2.3 e 2.4 representam duas hipóteses de configuração geradas através do configurador inicial, para que os ficheiros em PHP abram uma ligação para poder comunicar com os SGBDs, onde a primeira corresponde à ligação com o MySQL e a segunda com o PostgreSQL.

```php
<?php
    $serverName = "localhost";
    $userName = "administrator";
    $dbName = "mediaaccess";
    $pswd = "admin";
    $mysql = mysql_connect($serverName, $userName, $pswd) or trigger_error(mysql_error(),E_USER_ERROR);
    $db_selected = mysql_select_db($dbName, $mysql);
    $dbType = "mysql";
?>
```

Figura 2.3 - Exemplo de dados de configuração para abrir uma ligação entre o PHP e o MySQL

```php
<?php
    $serverName = "localhost";
    $userName = "administrator";
    $dbName = "mediaaccess";
    $pswd = "admin";
    $pg = pg_connect("host=$serverName dbname=$dbName user=$userName password=$pswd");
    $dbType = "psql";
?>
```

Figura 2.4 - Exemplo de dados de configuração para abrir uma ligação entre o PHP e o PostgreSQL

Como mencionado anteriormente os configuradores das componentes e as componentes não comunicam directamente com as bases de dados, pois o ActionScript não possui um mecanismo próprio para tal. Depois de correctamente configuradas, as componentes efectuam pedidos aos programas desenvolvidos em PHP, que fazem as consultas necessárias na base de dados para devolver a informação correcta às componentes.

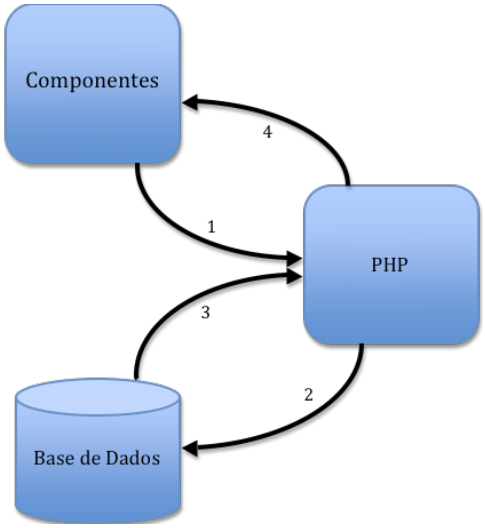

Figura 2.5 - Diagrama de comunicações entre os diferentes intervenientes

A Figura 2.5 exemplifica um fluxo de comunicação desde as componentes ou dos seus configuradores até à base de dados, de forma a que o utilizador possa interagir com o arquivo

multimédia. Para uma melhor compreensão do fluxo de comunicação, a Figura 2.5 dispõe de numeração para que seja possível seguir a sequência lógica que acontece quando uma das componentes pretende comunicar com o arquivo multimédia, aqui representado pelo seu elemento fundamental, a base de dados que armazena toda a informação sobre os conteúdos.

Como já foi dito, visto que as componentes necessitam de comunicar com o PHP, a linguagem de programação ActionScript contém um conjunto de objectos que permitem a comunicação com os programas desenvolvidos em PHP, sendo eles URLLoader, URLRequest e URLVariables.

```
var serverFile:String = "http://localhost/mediaaccess/php/createConfig.php";

var urlRequest:URLRequest = new URLRequest(serverFile);
var loader:URLLoader = new URLLoader();

var variables:URLVariables = new URLVariables();
variables.serverName = WebServerName.text;
variables.dbName = databaseName.text;
variables.connectionType = connectionBox.selectedItem.data;
variables.userName = userName.text;
variables.pass = pass.text;

urlRequest.data = variables;
urlRequest.method = URLRequestMethod.POST;

loader.addEventListener(Event.COMPLETE,serverResponse);

loader.load(urlRequest);

function serverResponse(evt:Event) {

    var variables:URLVariables = new URLVariables(loader.data);

    if (variables.resp == "false") {
        responseTxt.htmlText = "<b>Erro - <b/>Verifique as suas definições";
    }
    if (variables.resp=="true") {
        responseTxt.htmlText = "<b>Configuração concluída com sucesso<b/>";
    }
    loader.close();

}
```

Figura 2.6 - Exemplo de código que permite a comunicação entre ActionScript e o PHP

A Figura 2.6 representa um exemplo de código em ActionScript que permite de uma comunicação com o PHP. Nas primeiras linhas pode-se observar a declaração das variáveis necessárias, onde a primeira, do tipo URLRequest, é inicializada ao passar uma cadeia de caracteres que contém o endereço do ficheiro PHP que se pretende comunicar. Após isso é

declarada uma variável do tipo URLVariables que irá conter toda a informação que é passada para o PHP, através do método POST. No final existe uma variável do tipo URLLoader que, através do método *load*, inicia a comunicação com o endereço pretendido.

A diferença da comunicação entre ActionScript e o PHP, em comparação com as comunicações existentes nos sites convencionais, entre HTML e PHP, está na linha de código *loader.addEventListener(Event.COMPLETE, serverResponse);*. O ActionScript para além de orientado a objectos, funciona com base em eventos e esta linha de código indica que após o envio dos dados para o PHP terminar, deve ser executada a função *serverResponse*, dando a hipótese de obter uma resposta gerada no ficheiro PHP.

Na elaboração das componentes, os programas realizados em PHP apenas retornam cadeias de caracteres, contendo simples mensagens ou documentos XML obtidos através das consultas na base de dados. As consultas efectuadas na base de dados são feitas através das funções incluídas no PHP que utilizam os *sockets* de comunicação criados pelos SGBDs quando foram instalados.

No próximo capítulo, serão explicados os detalhes sobre o funcionamento das componentes realizadas durante a dissertação.

3. Implementação e Descrição das Componentes

Neste capítulo irão ser discutidas as opções tomadas na implementação das diferentes ferramentas e componentes. Em primeiro lugar serão abordada as ferramentas que dizem respeito às imagens e tudo o que se relaciona com elas, sendo que após isso haverá um secção correspondente ao vídeo. No final deste capítulo poderá ainda ser encontrado um secção dedicado à aplicação *desktop* desenvolvida.

3.1 Componente de Acesso a Imagens

No que diz respeito às imagens, foi desenvolvida uma ferramenta para construir uma componente que permitisse ao utilizador consultar as imagens existentes no arquivo multimédia. Visto que um dos objectivos desta dissertação é tornar as componentes independentes do arquivo multimédia, tornando-as reutilizáveis, esta ferramenta contribui para alcançar esse mesmo objectivo.

Tendo em conta um futura aplicação num contexto real, onde existe alguém responsável pela manutenção da página Web que permite o acesso ao arquivo, a ferramenta desenvolvida pode ser considerada como um configurador da componente em questão. Para utilizar este configurador a pessoa responsável por fazê-lo terá que ter um conhecimento mínimo da base de dados que serve de base ao arquivo multimédia.

Desta forma, o configurador comunica com o servidor Web que através do ficheiro criado durante a configuração inicial, mencionada no capítulo anterior, sabe qual o tipo de SGBD sobre a qual está criado o arquivo e pode tomar opções para a componente. Neste configurador, uma das hipóteses dadas é fazer a correspondência entre os dados que serão mostrados na componente e as tabelas e respectivos campos da base de dados. Ao comunicar com o servidor Web, este executa um programa realizado em PHP que retorna uma cadeia de caracteres que representa um

documento XML e que obedece ao XML Schema apresentado na Figura 3.1, com a informação sobre a base de dados.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="database">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="table" minOccurs="0" maxOccurs="unbounded" type="table"/>
        </xs:sequence>
        <xs:attribute name="name" use="required"></xs:attribute>
    </xs:complexType>
 </xs:element>
    <xs:complexType name="table">
        <xs:sequence>
            <xs:element name="field" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
            <xs:element name="key" minOccurs="1" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:simpleContent>
                        <xs:extension base="xs:string">
                            <xs:attributeGroup ref="keyAttributes"/>
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="name" type="xs:string"/>
    </xs:complexType>
    <xs:simpleType name="keyType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="primary"/>
            <xs:enumeration value="foreign_key"/>
            <xs:enumeration value="unique"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:attributeGroup name="keyAttributes">
        <xs:attribute name="type" type="keyType" use="required"/>
        <xs:attribute name="referenced_table" type="xs:string" use="optional"/>
        <xs:attribute name="referenced_column" type="xs:string" use="optional"/>
    </xs:attributeGroup>
 </xs:schema>
```

Figura 3.1 - XML Schema referente à base de dados que suporta o arquivo multimédia

Através do documento XML retornado pelo programa executado no servidor é possível que o responsável indique algumas propriedades da base de dados, como por exemplo a tabela que guarda a informação sobre as imagens e o campo dessa tabela que indica o título.

Desta forma, a ferramenta desenvolvida guia a pessoa responsável através de um processo de configuração, onde em primeiro lugar são configurados os aspectos técnicos da componente e após isso o aspecto visual. Na Figura 3.2 pode-se observar a ferramenta a pedir informação sobre a base de dados e na Figura 3.3 a possibilidade de configurar a disposição dos elementos da componente.

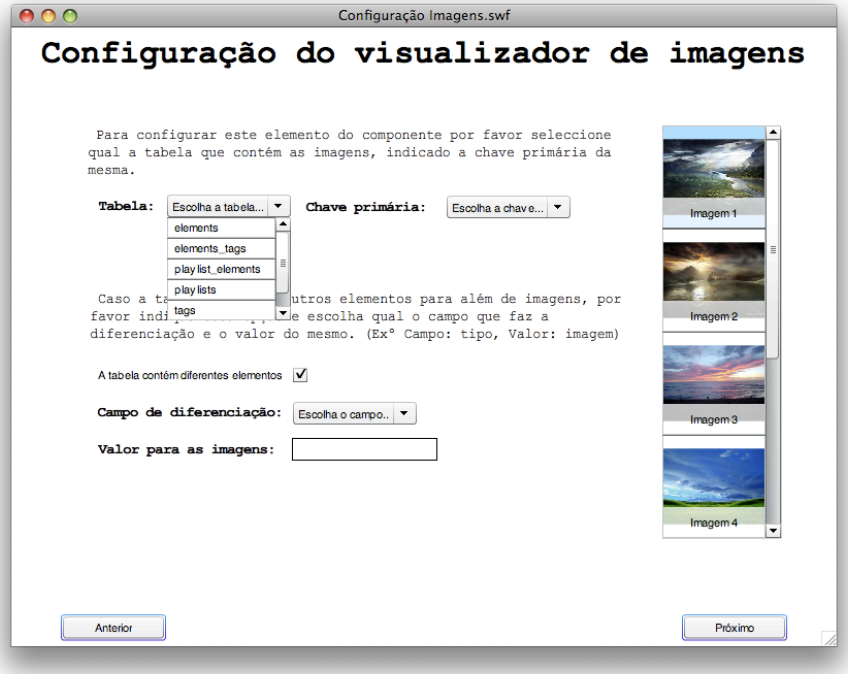

Figura 3.2 - Configurador a solicitar informação sobre a base de dados

Figura 3.3 - Configurador a permitir a escolha da localização dos elementos

Depois de efectuada a configuração, a informação é guardada no servidor de forma a que a componente resultante possa aceder aos conteúdos. Quando o configurador termina estas escolhas comunica com servidor Web que, ao receber os parâmetros referentes à base de dados,

produz um ficheiro PHP de forma dinâmica, cujo objectivo é ser utilizado pela componente. Ao ser executado pela componente, esse ficheiro PHP consulta a base de dados de forma a fornecer os dados necessários, como o título e a localização das imagens no servidor. Essa informação é passada através de um documento XML gerado pelo ficheiro PHP e enviado para a componente, obedecendo ao XML Schema apresentado na Figura 3.4.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="images">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="image" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="title">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:minLength value="0"/>
                                    <xs:maxLength value="25"/>
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:element>
                        <xs:element name="path">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:minLength value="1"/>
                                    <xs:maxLength value="100"/>
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:element>
                    </xs:sequence>
                    <xs:attribute name="id" type="xs:integer"></xs:attribute>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

Figura 3.4 - XML Schema referente aos dados das imagens

Já no que diz respeito à localização dos elementos na componente, essa informação é armazenada num ficheiro XML criado quando o configurador envia a informação sobre a disposição pretendida dos elementos e o título pretendido para essa componente. Esse documento, como todos os outros documentos XML usados na implementação, obedece às regras de um XML Schema, que neste caso é o apresentado na Figura 3.5.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="MA_ViewPic">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="title" minOccurs="1" maxOccurs="1">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:minLength value="23"/>
                            <xs:maxLength value="50"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
                <xs:element name="loader" minOccurs="1" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="loaderXPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="loaderYPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="list" minOccurs="1" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="listXPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="listYPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="horizontalList" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="controls" minOccurs="1" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="controlsXPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="controlsYPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="horizontalControls" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Figura 3.5 - XML Schema referente à localização dos elementos visuais da componente de imagens

Desta forma é possível à componente saber onde a pessoa responsável deseja que os elementos estejam posicionados, bem como obter as imagens existentes no arquivo multimédia. A utilização de documentos XML permite transferir a informação de uma forma simples e estruturada, tornando ao mesmo tempo fácil de detectar um problema de má formulação do documento. Para além destes factores, ao utilizar documentos XML exteriores à componente garante-se ainda mais independência do arquivo multimédia, visto que o documento que fornece informação sobre as imagens é gerado dinamicamente quando são feitos pedidos ao servidor.

Esta independência também se deve ao funcionamento das aplicações desenvolvidas em Flash, pois se de cada vez que o responsável pelas aplicações decidisse trocar o aspecto visual da componente, seria necessário editar o ficheiro flash e compilá-lo de forma a criar o ficheiro (*swf*) para que posteriormente possa ser incluído numa página Web ou executado no Flash Player *standalone*. Para além disso, caso a componente tivesse que ser utilizada num arquivo diferente

seria necessário procurar e alterar o ActionScript associado e mais uma vez compilar e produzir o ficheiro (*swf*) para que a componente pudesse ser utilizada. Assim ao ter uma ferramenta que permite efectuar a configuração para o arquivo pretendido, é possível fazer alterações quer a nível do arquivo, quer a nível do aspecto visual sem que seja necessário recompilar o ficheiro criado no Adobe Flash.

A componente tem um aspecto simples para que possa ser inserido em diferentes páginas Web sem influenciar em demasia o design da página, mas permitindo que o utilizador tenha acesso aos conteúdos de forma eficaz. No final o resultado da aplicação, após realizar a configuração, é igual ao apresentado na Figura 3.6.

Figura 3.6 - Aspecto final da componente de imagens após configurada

3.2 Componente de Acesso a Vídeos

À semelhança do que acontece com as outras componentes desenvolvidas, que necessitam de ferramentas que permitam a sua configuração, as que utilizam vídeos também necessitam de ser configuradas. Tal como acontece na componente de visualização de imagens, referida no capítulo 3.1, esta componente é configurada no que diz respeito aos elementos que necessita de consultar na base de dados, bem como ao seu aspecto visual.

As informações obtidas durante a configuração são armazenadas de duas formas diferentes. A informação sobre a sua aparência é armazenada num ficheiro XML que respeita o Schema apresentado na Figura 3.7, ao passo que para obter informação da base de dados é gerado dinamicamente um ficheiro PHP que é consultado quando a componente é executada.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="MA_ViewVideo">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="title" minOccurs="1" maxOccurs="1">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:minLength value="22"/>
                            <xs:maxLength value="50"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
                <xs:element name="loader" minOccurs="1" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="startX" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="startY" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="list" minOccurs="1" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="listXPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="listYPos" type="xs:integer" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="horizontalList" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Figura 3.7 - XML Schema referente à aparência da componente que reproduz os vídeos

Este documento XML permite que a componente quando começa a ser executada, possa dispor os seus elementos de acordo com o que foi escolhido. Para além de consultar este documento XML, a componente adquire informação sobre os vídeos através de outro documento XML, obtido através da consulta ao ficheiro PHP gerado através do configurador. Desta forma este último documento XML, que respeita o Schema parcialmente apresentado na Figura 3.8, fornece as informações sobre a localização dos vídeos, bem como o título dado pelo utilizador. O aspecto final da componente que permite a reprodução de vídeos pode ser observado na Figura 3.9.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="videos">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="video" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="title" minOccurs="1" maxOccurs="1"/>
                            <xs:element ref="path" minOccurs="1" maxOccurs="1"></xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:integer" use="required"/>
                    </xs:complexType>
                    <xs:key name="videoPK">
                        <xs:selector xpath="video"></xs:selector>
                        <xs:field xpath="@id"></xs:field>
                    </xs:key>
                </xs:element>
                <xs:element name="editedVideo" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="title" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="editedElement" minOccurs="1" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="order" type="xs:positiveInteger" minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="startTime" type="xs:decimal" minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="stopTime" type="xs:decimal" minOccurs="1" maxOccurs="1"/>
                                        <xs:element ref="path" minOccurs="1" maxOccurs="1"/>
                                    </xs:sequence>
                                    <xs:attribute name="vidId" type="xs:integer" use="required"/>
                                </xs:complexType>
                                <xs:keyref name="editedVideoFK" refer="videoPK">
                                    <xs:selector xpath="editedVideo"/>
                                    <xs:field xpath="@vidId"/>
                                </xs:keyref>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="mixId" type="xs:integer" use="required"></xs:attribute>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:simpleType name="videoTypeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="original"/>
            <xs:enumeration value="mix"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:element name="title">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:minLength value="0"/>
                <xs:maxLength value="25"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>

    <xs:element name="path">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:minLength value="1"/>
                <xs:maxLength value="100"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>

</xs:schema>
```

Figura 3.8 - XML Schema referente aos dados dos vídeos

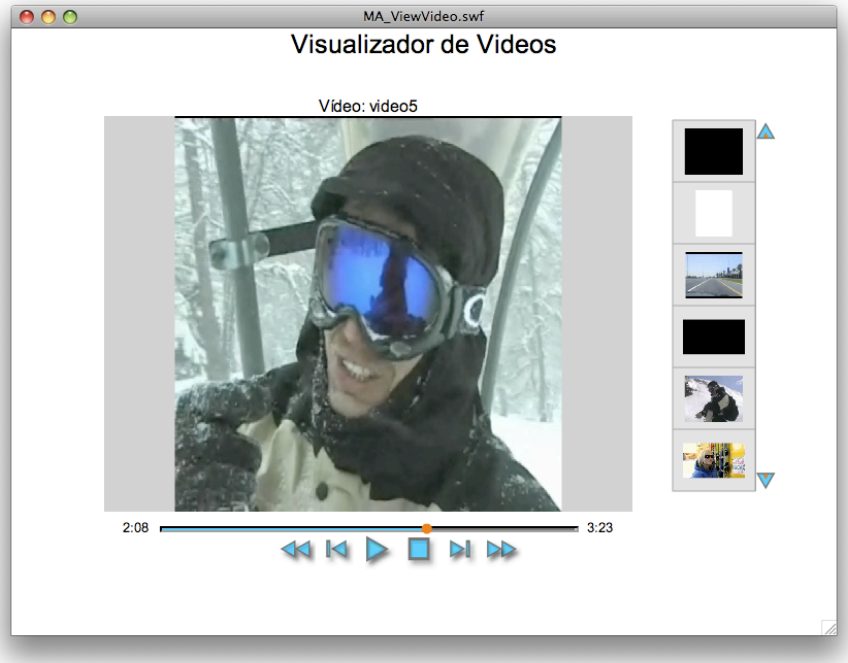

Figura 3.9 - Aspecto final da componente que reproduz os vídeos

A componente é composta por 3 elementos, sendo eles uma lista de vídeos, a zona de visualização, onde é apresentado o vídeo e os controlos do vídeo. Quase todos estes elementos poderiam ser encontrados no Adobe Flash, mas para uma maior independência e um maior controlo sobre o vídeo e os elementos apresentados, foram desenvolvidas, em ActionScript, algumas classes. Para uma melhor compreensão das classes e dos seus objectivos, de seguida é apresentado um resumo das mesmas:

- **MA_ButtonEvent:** esta classe é uma extensão da classe Event existente na linguagem ActionScript 3.0 e é utilizada para gerar eventos ligados aos botões de controlo existentes na componente.
- **MA_ButtonControl:** nesta classe são controlados os botões de controlo e de acordo com os que são pressionados, é gerado um evento do tipo MA_ButtonEvent que será interpretado pelo leitor;
- **MA_VideoEvent:** tal como a classe MA_ButtonEvent, mencionada anteriormente, esta classe é uma extensão à classe Event, sendo que a classe MA_VideoEvent permite gerar eventos relacionados com o vídeo, por exemplo indicar qual o vídeo a ser reproduzido.

- **MA_Thumb:** classe utilizada para criar as miniaturas apresentadas na lista de vídeos. Esta classe, para além de permitir criar elementos para preencher a lista ao passar o rato por cima da miniatura, inicia uma pré-visualização do vídeo correspondente. Para esta funcionalidade é necessário carregar o vídeo em questão.
- **MA_ThumbList:** esta classe foi pensada de forma a que seja possível criar uma lista horizontal ou vertical com um número configurável de linhas e colunas, sendo possível conter várias páginas de miniaturas. Tendo em conta que as miniaturas permitem efectuar uma reprodução do vídeo, é necessário ter em atenção que quanto maior for o número de miniaturas, maior será o tempo de espera até que estas se encontrem preparadas.
- **MA_Player:** Esta é a classe principal que adiciona à componente a zona de visualização do vídeo e interpreta todos os eventos relacionados com a reprodução do vídeo, ou seja, esta classe é responsável pela correcta reprodução do vídeo.

Como se pode verificar pela observação da Figura 3.8, este reprodutor permite não só visualizar os vídeos na sua forma original, mas também reproduzir uma lista de vídeos, que respeita uma ordem imposta pelo utilizador, onde o tempo de início e de fim de cada elemento possa ter sido alterado. Para que o utilizador possa efectuar esta edição, existe uma ferramenta para configurar esta componente de edição. Nesta ferramenta, ao contrário das apresentadas anteriormente, não existe um configuração do aspecto visual. Neste caso o responsável pelas configurações apenas configura o que está relacionado com a base de dados, indicando a tabela de vídeos originais e a tabela de vídeos editados, bem como os restantes campos necessários.

Nesta ferramenta é assumida que a construção da base de dados tem em conta alguns aspectos, como por exemplo uma tabela auxiliar entre a tabela de vídeos originais e vídeos editados, uma vez que a relação entre estas será do tipo many-to-many (N-N) [24]. Desta forma, quando a pessoa responsável pela configuração indica as duas tabelas, a ferramenta consulta o documento XML que indica as tabelas da base de dados (ver Figura 3.1) e sugere qual a tabela auxiliar, sendo que esta pode depois ser alterada. Esta sugestão é feita através da pesquisa no documento XML, tentando encontrar uma tabela que contenha duas chaves externas, em que uma referencia a tabela de vídeos originais e a outra a tabela de vídeos editados. Após isto a

componente de edição está pronta a ser disponibilizada ao utilizador, tendo o aspecto final mostrado na figura 3.10.

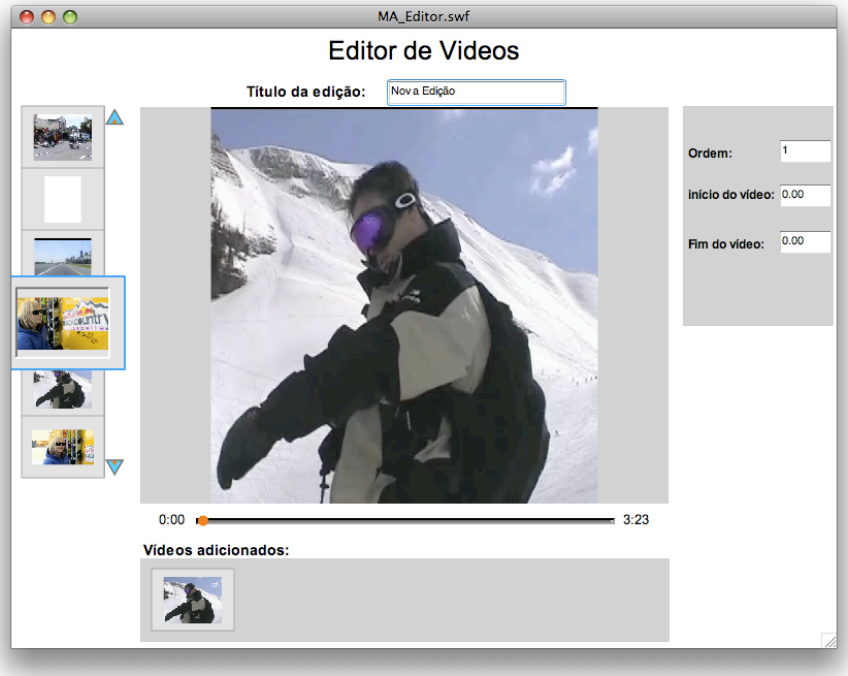

Figura 3.10 - Aspecto final da componente que permite a edição dos vídeos

Deve ainda ser mencionado que as ferramentas e componentes mencionadas quer neste capítulo, quer no anterior, ao início pedem o endereço do site em que vão ser executadas, para que contactem um ficheiro PHP, denomindado *ping.php* que conforme a componente ou ferramenta que efectua a chamada devolve o endereço para os restantes ficheiros desenvolvidos em PHP necessários para que tudo funcione correctamente.

3.3 Aplicação Desktop para Upload de Conteúdos

De acordo com o que foi dito anteriormente, um dos objectivos do trabalho consiste em criar componentes para permitir o acesso a conteúdos multimédia e também contribuir para o aumento desse mesmo arquivo através da participação dos utilizadores.

Esta componente foi desenvolvida para permitir que os utilizadores possam contribuir para o aumento do arquivo multimédia. É comum encontrar este tipo de situações em sites onde

os utilizadores partilham conteúdos entre si, efectuando *upload* de conteúdos através de uma página Web. No entanto esta componente não foi desenvolvida com o intuito de ser integrada numa página Web, pois é necessário efectuar uma instalação da mesma no computador do utilizador e utilizar o ambiente de execução Adobe AIR [25].

Este ambiente de execução faz com que esta componente seja multi-plataforma, pois é possível executar o AIR em Windows (2000, XP ou Vista), Mac OS X (10.4 ou superior) e Linux. Desta forma, independentemente do sistema operativo utilizado um utilizador do arquivo multimédia poderá fazer o *upload* dos seus ficheiros de imagem ou vídeo, dentro dos tipos suportados pelo arquivo.

No que diz respeito ao desenvolvimento, para além do AIR, foi necessário obter de forma gratuita o SDK do Adobe Flex [26], que já inclui o SDK do Adobe AIR [27]. O Flex como já foi referido na secção 1.4.2, recorre a uma linguagem chamada MXML, baseada no XML e ActionScript. A Figura 3.11 mostra um exemplo muito simples de um ficheiro *mxml* onde se verifica não só a semelhança da linguagem MXML com o XML, mas também a presença de código em ActionScript. Na figura 3.12 é apresentada a aplicação resultante deste código após a sua compilação.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" width="300" height="200" creationComplete="load();">

    <mx:Script><![CDATA[
        import mx.controls.Alert;

        public function load():void{
            textArea.text = "Clique no botão para mostrar um alerta";
        }


        private function showAlert(event:Event):void {
            var alert:Alert = Alert.show("Obrigado por carregar no botão");
        }
    ]]>
    </mx:Script>

    <mx:TextArea id="textArea" width="150" height="100">
    </mx:TextArea>
    <mx:Button id="button" label="Mostrar alerta" click="showAlert(event)" />
</mx:Application>
```

Figura 3.11 - Exemplo de código MXML em conjunto com ActionScript

Figura 3.12 - Aplicação resultante do código apresentado na Figura 3.11

O SDK do Flex permite criar aplicações Web que podem ser executadas em *browsers* através do *plugin* do Flash Player, ou na sua versão *standalone*, mas como já foi referido, também permite criar aplicações *desktop* que apenas podem ser executadas recorrendo ao ambiente de execução Adobe AIR. As maiores diferenças na criação destes dois tipos de aplicações são no código e a forma como este é compilado.

Para identificar que se pretende uma aplicação *desktop*, ao contrário do código apresentado na Figura 3.11 onde a *tag* que identifica a aplicação é do tipo:

<mx: Application > </mx:Application>

é necessário fazer uma ligeira alteração e utilizar a *tag*:

<mx:WindowedApplication > </mx:WindowedApplication>

Para além desta alteração, a compilação que normalmente recorre ao comando *mxmlc*, terá que ser feita através do comando *amxml* que é um atalho existente para o comando *mxmlc* utilizando as *flags* necessárias para que o resultado da compilação seja um ficheiro preparado para executar em Adobe AIR. No caso das aplicações normais o processo de criação da aplicação estaria concluído, sendo que o ficheiro com a extensão *swf* ficaria criado e pronto a ser utilizado, mas no caso das aplicações de desktop, e tendo em conta que este necessita de ser instalado no computador do utilizador, é necessário executar mais alguns passos para garantir a segurança do sistema.

Após a criação do ficheiro *swf* é necessário criar um ficheiro do tipo *air* para que de seguida se possa finalmente instalar a aplicação. Para criar ficheiro de instalação é necessário ter um certificado, ou criar um certificado próprio através do comando:

adt -certificate -cn name [-ou org_unit][-o org_name][-c country] key_type pfx_file password

onde o importante são os parâmetros:

- ***name*:** correspondente ao nome do certificado;
- ***key_type*:** aqui é onde se indica o tipo de chave que protege o certificado, dentro das duas hipóteses existentes, 1024-RSA ou 2048-RSA;
- ***pfx_file*:** este parâmetro serve para definir o nome do ficheiro físico que corresponde ao certificado, que deve ter a extensão p12;
- ***password*:** cadeia de caracteres que serve de palavra-chave do certificado e necessária para assinar o ficheiro *air* que irá ser criado posteriormente;

Para finalizar é necessário executar um último comando que cria o pacote de instalação devidamente assinado e seguro para o utilizador,

adt –package -storetype pkcs12 -keystore pfx_file air_file app_xml [file_or_dir | -C dir file_or_dir | -e file dir ...]

onde os parâmetros correspondem ao seguinte:

- **pfx_file:** ficheiro físico do certificado, criado pelo comando anteriormente citado;
- **air_file:** nome pretendido para o pacote de instalação, que deve conter a extensão *air*;
- **app_xml:** parâmetro que corresponde a um ficheiro XML, denominado por *file descriptor*, que contém algumas propriedades auxiliares à criação do pacote auxiliar. Pode-se observar o exemplo do ficheiro XML correspondente à aplicação desenvolvida na Figura 3.13.
- ***[file_or_dir | -C dir file_or_dir | -e file dir ...]:*** neste parâmetro, ou parâmetros, são indicados os ficheiros, ou a directoria, necessários para o correcto funcionamento da aplicação.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://ns.adobe.com/air/application/1.0">
    <id>FileUploader</id>
    <version>1.0</version>
    <filename>FileUploader</filename>
    <name>MediaAccess - FileUploader</name>
    <description>
        Programa para fazer upload de conteúdos para o MediaAccess
    </description>
    <copyright>Copyright © 2009 IMG-CITI-UNL</copyright>
    <initialWindow>
        <content>FileUploader.swf</content>
    </initialWindow>
    <installFolder>MediaAccess</installFolder>
</application>
```

Figura 3.13 - Exemplo do *file descriptor* da aplicação desenvolvida

Depois de todo este processo de compilação e criação do pacote de instalação, já é possível instalar a aplicação. Na figura 3.14 e 3.15 é possível observar a parte do processo de instalação da aplicação *desktop* MediaAccess – FileUploader, onde se permite a instalação e se decide o destino da instalação.

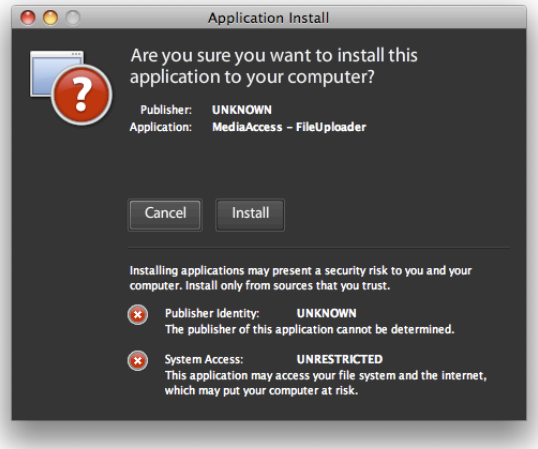

Figura 3.14 - Permissão para instalar a aplicação desktop desenvolvida

Figura 3.15 - Escolha da localização para a instalação da aplicação desenvolvida

Contando com a aplicação já instalada, o utilizador pode desfrutar da mesma para enviar conteúdo para o arquivo multimédia. A aplicação, durante a sua execução, é apresentada na Figura 3.16, podendo fazer uma divisão da aplicação em três partes lógicas.

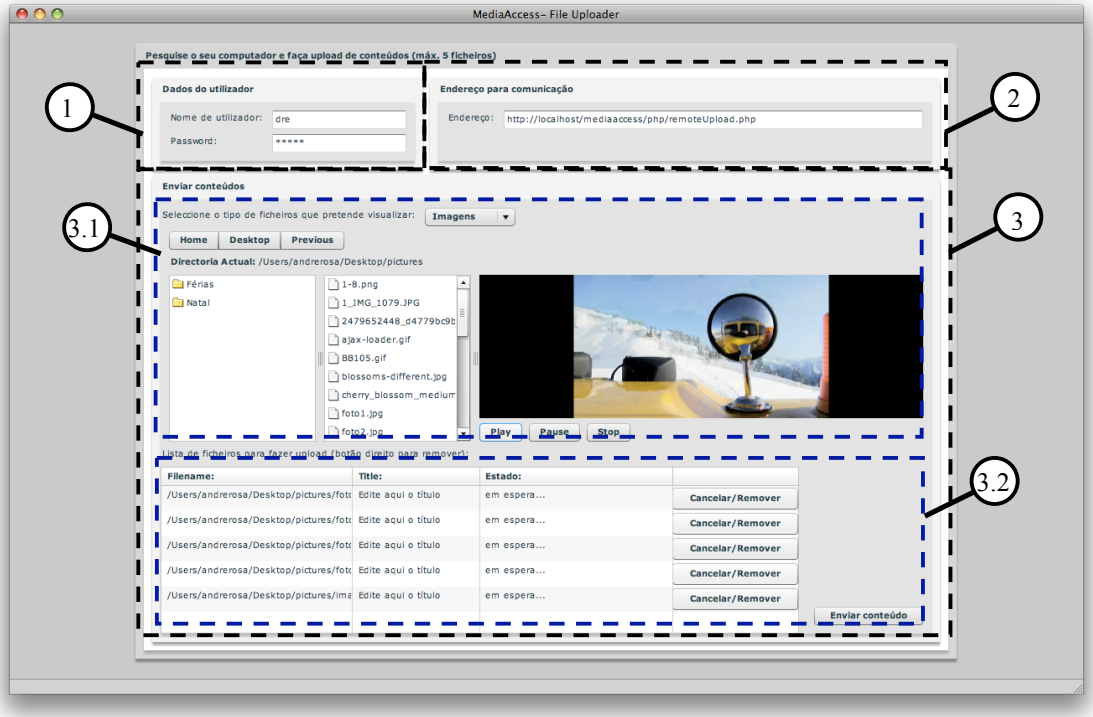

Figura 3.16 - Aplicação *desktop* MediaAccess – FileUploader a ser executada

As partes que constituem a aplicação têm as seguintes funcionalidades:

1 - Nesta zona da aplicação o utilizador insere os seus dados, nome de utilizador e *password*, que lhe permitem efectuar o *login* e posteriormente enviar os conteúdos multimédia;

2 - Aqui o utilizador insere o endereço que pretende que a aplicação comunique. Num contexto real, como o apresentado no capítulo 4, este endereço poderia ser fornecido quando fosse feito o *download* da aplicação. Uma vez inserido, este valor é guardado num ficheiro nos documentos do utilizador para evitar que o utilizador esteja sempre a inseri-lo;

3 - Nesta secção o utilizador pode explorar o seu computador e escolher os ficheiros que pretende enviar, bem como manter um histórico do enviado.

3.1 - Nesta subsecção, para além de explorar o computador, o utilizador pode filtrar os ficheiros que lhe são apresentados, escolhendo entre imagens, vídeo e ambos. Quando o utilizador selecciona um dos ficheiros apresentados, obtém uma pré-visualização da imagem ou vídeo em questão. Após confirmar se é esse ficheiro que pretende enviar, o utilizador efectua um duplo clique sobre o ficheiro, que automaticamente passa para a lista de ficheiros no fundo da aplicação;

3.2 - É nesta zona, onde se encontra a lista de ficheiros a enviar e simultaneamente os ficheiros já enviados que o utilizador pode enviar os ficheiros propriamente ditos. O utilizador, depois de ter a lista populada, pode então pressionar no botão "Enviar conteúdo" para proceder ao envio. Esta lista mantém o utilizador informado do estado em que se encontram os ficheiros ("Em espera", "A enviar", "Envio completo" ou "Erro ao enviar conteúdo"), permitindo desta forma visualizar o histórico dos ficheiros enviados, bem como os que estão em espera para enviar.

Como mencionado anteriormente, esta aplicação permite explorar o computador do utilizador e visualizar os seus ficheiros, recorrendo à API do Adobe AIR, mais concretamente às classes FileSystemList e File. A primeira classe permite que seja apresentada uma lista de

ficheiros do computador do utilizador, ao passo que a classe File faz com que seja possível manipular os ficheiros individualmente.

A classe File contém o método *upload*, que em conjunto com as classes URLRequest e URLVariables, permite enviar o ficheiro para o servidor, recebendo após isso uma resposta do programa que se encontra no lado do servidor. A resposta dada pelo servidor permite informar o utilizador sobre resultado do envio.

Desta forma, recorrendo a esta aplicação o utilizador do arquivo multimédia poderá enviar os seus conteúdos, sem que seja necessário o executar o *browser* e ter que esperar pelos tempos de carregamento das páginas Web.

4. Integração das Componentes

Para verificar o funcionamento das componentes num contexto real foi desenvolvido um site que englobasse as componentes, bem como algumas das tecnologias mais relevantes no desenvolvimento para a Web.

O site é constituído por páginas Web desenvolvidas em HTML e PHP, sendo esta última tecnologia utilizada para quando existe a necessidade de comunicar com uma base de dados, como por exemplo para efectuar o registo de um utilizador ou o *login* no site. Para além destas duas tecnologias foram utilizadas folhas de estilo CSS para definir o aspecto visual de todo o site.

Para que seja possível iniciar a sessão é preciso fornecer o nome de utilizador e a password, permitindo ao utilizador visualizar as restantes páginas do site. Para além das opções normais, caso o utilizador tenha privilégios de administrador irá ter a opção de aceder aos configuradores das componentes, funcionando como *Back Office*. Para um melhor entendimento da estrutura do site a Figura 4.1 apresenta o mapa correspondente.

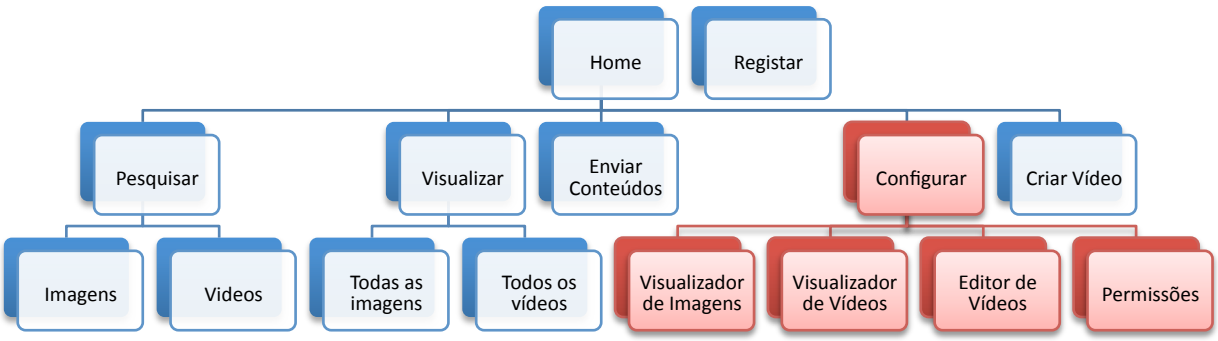

Figura 4.1 - Mapa do site construído para testar componentes

Quando alguém pretende obter privilégios de administrador, terá que ser sempre outra pessoa com esses privilégios a dá-los, pois isso não é uma opção que se escolha na altura do registo. Claro que o primeiro utilizador a beneficiar desses privilégios, terá que obtê-los a partir

da alteração directa da base de dados por parte do administrador, mas após isso estará apto para o fazer com outros utilizadores. Esta opção tem que ser usada com cuidado, para que não sejam dados privilégios a pessoas indesejadas que poriam em causa o bom funcionamento das componentes e de todo o site.

Neste caso, para implementar o exemplo descrito neste capítulo, bem como testar o funcionamento das componentes, individualmente ou integradas numa página Web, foi necessário desenvolver também uma base de dados para suportar o arquivo. Para esse efeito foram utilizados o MySQL e o PostgreSQL. A Figura 4.2 apresenta um esquema da base de dados para uma melhor compreensão do que foi desenvolvido.

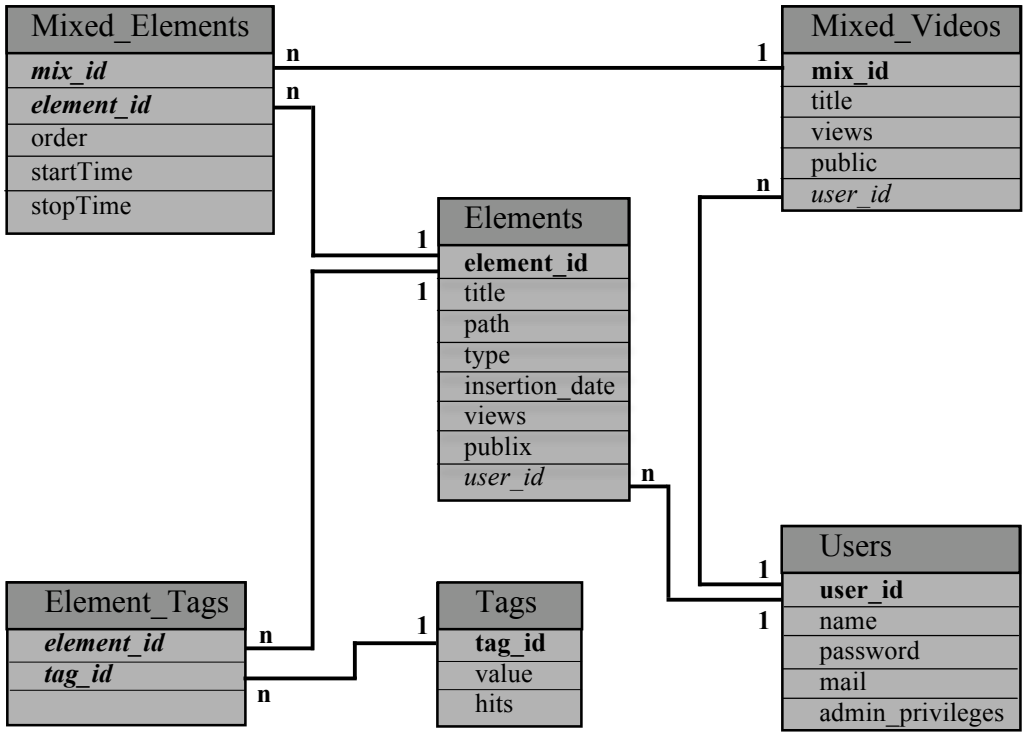

Figura 4.2 - Esquema da base de dados da aplicação

O *upload* de conteúdos para este arquivo pode ser efectuado a partir da aplicação *desktop* explicada no capítulo anterior, ou a partir de uma página de *upload* de conteúdos acessível a partir do site desenvolvido. A aplicação *desktop* foi criada para que o utilizador enviasse os conteúdos, sem ter que inserir palavras-chave que servem de metadados para ajudar na pesquisa dos conteúdos, enquanto que a versão existente no site dá essa possibilidade. Para além disso, a

versão *desktop* destina-se a um utilizador que no momento em que pretende inserir conteúdo não quer perder tempo a navegar ou à espera que os conteúdos das páginas.

Outro exemplo do uso conjunto entre as componentes desenvolvidas em Flash e o PHP corresponde às pesquisas, pois as palavras chave são enviadas e processadas através do PHP e após isso, quando a componente pede as informações ao servidor utiliza programas desenvolvidos em PHP, para obter a informação tendo em conta a pesquisa efectuada. Para esta situação são utilizadas as variáveis de sessão disponibilizadas pelo PHP e acessíveis através da instrução $_SESSION[‘<nome da variável>’], que neste caso irá conter um conjunto de palavras chave.

A Figura 4.3 mostra a página inicial onde é possível efectuar o login ou ir para a página de registo, já as Figuras 4.4 e 4.5 mostram as componentes integradas no site. A primeira corresponde a uma componente de visualização e a segunda a uma componente de configuração.

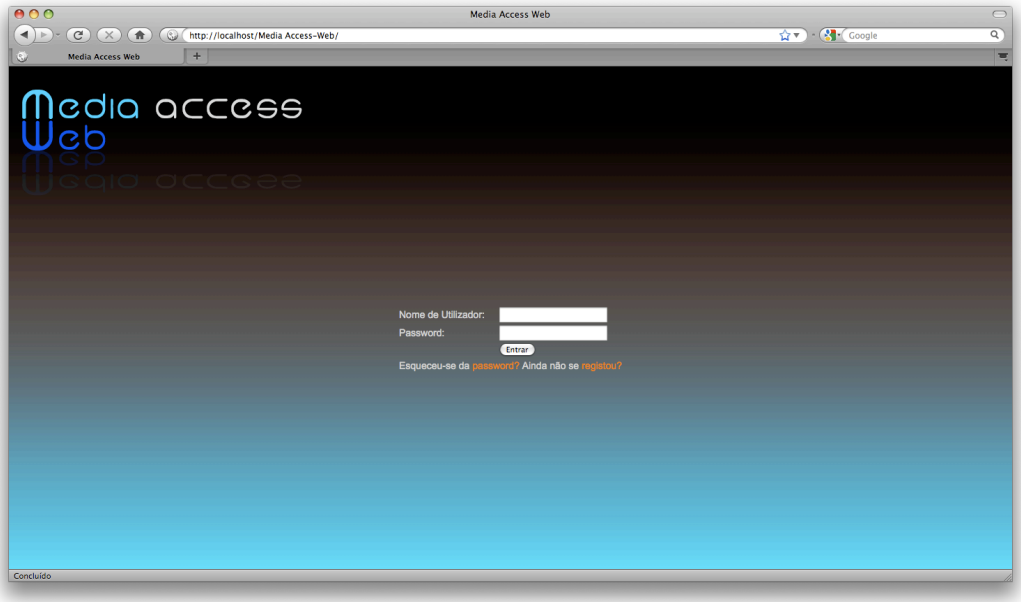

Figura 4.3 – Início do site onde é pedido para efectuar o login

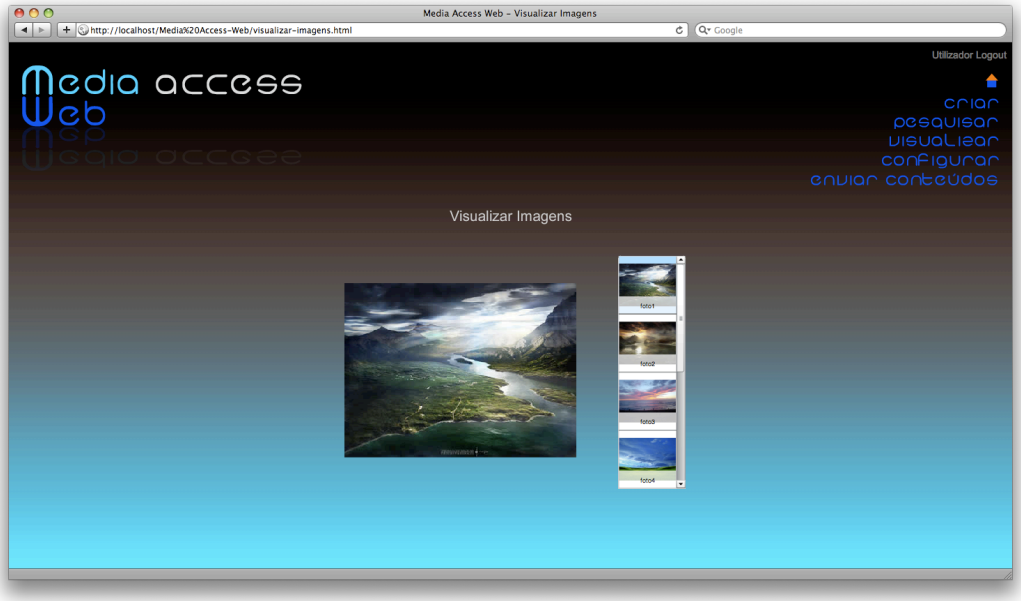

Figura 4.4 - Exemplo do site com uma componente de visualização incluída

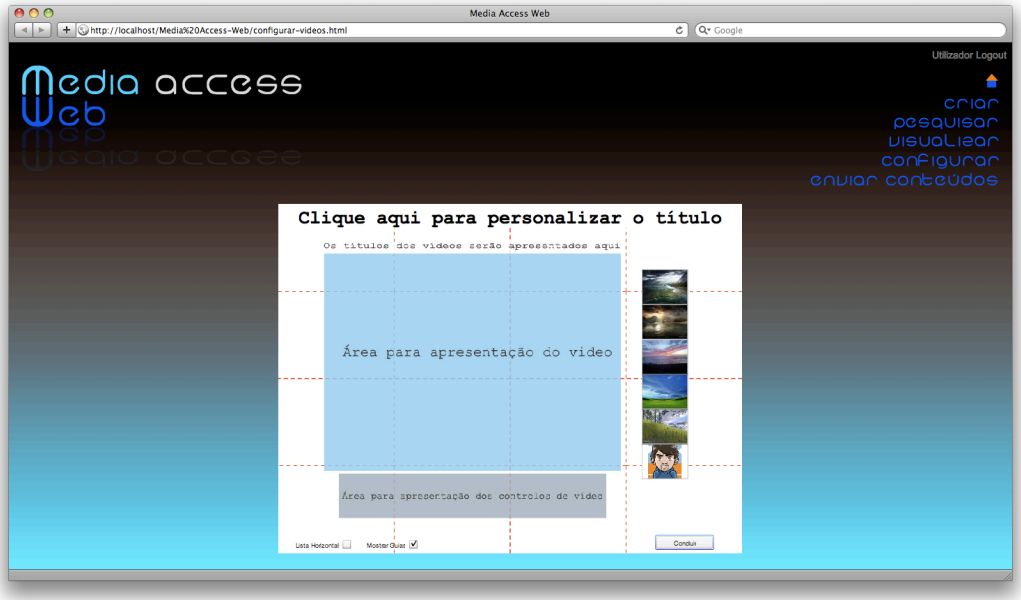

Figura 4.5 - Exemplo do site com uma componente de configuração incluída

5. Conclusões e Trabalho Futuro

No presente capítulo será apresentada uma avaliação critica ao trabalho realizado, mencionando não só as ferramentas e componentes desenvolvidas, mas também as tecnologias utilizadas para o atingir o objectivo final. Após isso serão mencionados alguns pontos que poderão ser explorados num trabalho futuro em que o ponto de partida seja o resultado desta dissertação.

5.1 Avaliação Crítica

Através do estado da arte relacionado com o tema desta dissertação, podemos verificar que os arquivos multimédia que disponibilizam os conteúdos para utilizadores no geral, servem-se de ferramentas construídas especificamente para cada caso. No que diz respeito à Web, isso também se verifica nos sites que permitem o acesso ao conteúdo multimédia onde, para cada site, se torna necessária a construção de um site que se adapte a situação em causa. Para além disso, quando ocorrem alterações ou reestruturações que implicam a base desses sites, há que percorrer as linhas de código dos ficheiros (HTML, PHP, JSP e afins) que constituem os mesmos.

É neste ponto que incide o trabalho realizado, promovendo a construção de ferramentas que permitam a criação de componentes reutilizáveis em diferentes contextos. No que diz respeito à reutilização, tendo em conta as tecnologias utilizadas e mencionadas anteriormente, foi possível obter uma independência dentro de um certo limite, que permite utilizar as componentes em diferentes situações. Em parte, esta característica é alcançada devido ao facto de as ferramentas acederem às propriedades mais genéricas da base de dados, na qual assenta o arquivo multimédia.

O desenvolvimento de um site para testar as componentes, explicado no capítulo 4, demonstrou que as componentes, através da sua correcta configuração, permitem o acesso aos

conteúdos num contexto real. Apesar de, para este exemplo, o número de conteúdos não seja em grande escala, as componentes conseguem obter a informação desejada e apresentar os resultados. É possível que hajam algumas limitações neste aspecto, nomeadamente na componente que permite visualizar as imagens, com o elemento que apresenta as imagens, se o número for muito elevado.

No que diz respeito às tecnologias utilizadas e uma vez que a experiência prévia quer em Flash, Flex e ActionScript era nula, torna-se possível avaliar a facilidade com que se aprendem e se desenvolvem aplicações com estas tecnologias. Durante o decorrer da aprendizagem do funcionamento das ferramentas da Adobe torna-se evidente que devido à semelhança, do MXML com o XML, o SDK do Flex destaca-se quando apresentado a uma pessoa com experiência anterior em programação. Torna-se mais fácil de entender o funcionamento das *tags* adoptadas pela linguagem e os resultados aparecem mais facilmente. Mesmo após obter um resultado bastante genérico e pouco elaborado a nível visual é possível melhorá-lo recorrendo a folhas de estilo CSS.

Esta facilidade de adaptação ao Flex, não significa que o Flash seja posto de parte, mas nesta ferramenta é necessário uma maior insistência na aprendizagem, pois para obter um melhor resultado são necessários algumas noções de design. Para além de ser uma ferramenta muito virada para o manuseamento visual dos elementos nela existentes, também é necessário compreender a existência de um elemento temporal, a Timeline.

Graças ao ActionScript e às suas parecenças com outras linguagens orientadas a objectos, como por exemplo o Java, a sua aprendizagem é bastante intuitiva, permitindo após isso um enriquecer as aplicações desenvolvidas quer em Flash quer em Flex. Na verdade é a utilização desta linguagem que permite optimizar a interacção e funcionamento das aplicações desenvolvidas com as ferramentas da Adobe.

5.2 Trabalho Futuro

Considerando a possibilidade desta dissertação servir de ponto de partida para algum trabalho e tendo em conta o que foi desenvolvido durante esta dissertação, pode-se notar algumas limitações que poderiam aperfeiçoadas num trabalho futuro.

Para que o leque de opções fornecidas pelas componentes já desenvolvidas pudesse aumentar, as ferramentas que servem de base para a sua construção poderiam ser modificadas de forma a dar novas hipóteses de configuração. Um dos exemplos mais simples seria a possibilidade de alterar mais aspectos visuais, como as fontes ou a cor da escrita existente nas componente. Desta forma seria interessante a possível colaboração com uma pessoa ligada à área de Design, visto que assim seriam aproveitados os conhecimentos e experiências ao nível da componente visual.

Visto que os conteúdos multimédia abrangidos pelas componentes desenvolvidas são o vídeo e a imagem, seria interessante desenvolver componentes para o áudio, tornando possível não só a sua reprodução, mas também a criação de listas de reprodução das músicas integrais ou editadas.

Como foi demonstrado anteriormente, as aplicações *desktop* fazem parte de uma nova vaga de aplicações Web, que poderia ser explorada dando seguimento a este trabalho e elaborando uma versão *desktop* de uma aplicação que incluísse todas as componentes ou apenas as desejadas.

Para finalizar, quando todos os tipos de conteúdos multimédia estivessem abrangidos pelas componentes, quando os configuradores e as componentes fornecessem mais opções, seria de extremo interesse implementar todo o sistema num arquivo multimédia real e com um maior volume de conteúdos, disponibilizando aos utilizadores uma nova forma de acesso aos conteúdos multimédia. Através da aplicação num arquivo multimédia de grande dimensão, seria possível obter mais resultados sobre a capacidade de resposta e apresentação dos conteúdos, podendo nessa altura ser necessário efectuar algumas alterações na forma como os resultados são apresentados e evitar a carga excessiva das componentes.

6. Bibliografia

[1] J. R. Smith, **Digital Vídeo Libraries and the Internet**. *IEEE Communications Magazine*, Vol. 37, p. 92-97, 1999.

[2] M. Christel, H. Wactlar, S. Stevens, M. Sirbu, R. Reddy, M. Mauldin, T. Kanade, **Informedia Digital Video Library**. *Communications of the ACM*, Vol. 38, No 4, p. 57-58, 1995.

[3] H. Wactlar, **Informedia: Search and Summarization in the Video Medium**. *Imagina 2000 Conference*, Mónaco, 2000.

[4] M. Dönderler, E. Saykol, Ö. Ulusoy, U. Güdükbay, **BilVideo: A Video Database Management System**. *IEEE Multimedia*, Vol. 10, p. 66-70, 2003.

[5] M. Dönderler, E. Saykol, U. Arslan. Ö. Ulusoy, U. Güdükbay, **BilVideo: Design and Implementation of a Video Database Management**. *System Multimedia Tools and Applications*, Vol. 27, p. 79–104, 2005.

[6] J. Casares, **SILVER: An Intelligent Video Editor**. *Conference on Human Factors in Computing Systems (ACM CHI'01)* - Student Posters, p. 425-426, Seattle (Washington, E.U.A.), 2001.

[7] B. Myers, J. Casares, S. Stevens, L. Dabbish, D. Yocum, A. Corbett, **A Multi-View Intelligent Editor for Digital Video Libraries**. *First ACM/IEEE Joint Conference on Digital Libraries (JCDL'01)*, p. 106-115, Roanoke (Virginia, E.U.A), 2001.

[8] Photobucket, **http://photobucket.com**. *Online*, última visita em 28 de Janeiro de 2009

[9] Yahoo! Research, **Content, Metadata, and Behavioral Information: Directions for Yahoo! Research**. *IEEE Data Engineering Bulletin*, Vol. 29, No 4, p.10-18, 2006.

[10] R. Baeza-Yates, A. Tiberi, **Extracting Semantic Relations from Query Logs**. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07)*, p. 76-85, San José (Califórnia, E.U.A), 2007.

[11] J.M. Martinez, R. Koenen, F. Pereira, **MPEG-7: The generic multimedia content description standard, part 1**. *IEEE Multimedia*, Vol. 9, p. 78-87, 2002.

[12] I. Burnett, R. Van de Walle, K. Hill, J. Bormans, F. Pereira, **MPEG-21: Goals and Achievements**. . *IEEE Multimedia*, Vol. 10, p. 60-70, 2003.

[13] B. L. Tseng, C. Lin, J. R. Smith, **Using MPEG-7 and MPEG-21 for Personalizing Video**. *IEEE Multimedia*, Vol. 11, p. 42 -52, 2004.

[14] R. Jesus, A. Abrantes, N. Correia, **Photo Retrieval from Personal Memories using Generic Concepts**. *Advances in Multimedia Information Processing - PCM 2006*, Vol. 4261, p. 633-640, Hangzhou (China), 2006.

[15] Adobe Flash Platform, **http://www.adobe.com/flashplatform/**. O*nline*, última visita em 28 de Julho de 2009.

[16] Adobe AIR: Browser vs Desktop, **http://www.adobe.com/products/air/comparison/**. O*nline*, última visita em 28 de Julho de 2009.

[17] Extensible Markup Language (XML), **http://www.w3.org/XML/**. *Online*, última visita em 28 de Julho de 2009.

[18] Jesse J. Garrett, **Ajax: A New Approach to Web Applications**. *Adaptive Path - Essays (http://www.adaptivepath.com/publications/essays/archives/000385.php)*, 2005.

[19] Youtube – Broadcast Yourself, **www.youtube.com**. *Online*, última visita em 28 de Julho de 2009.

[20] Cuong Do, **Youtube Scalability**. *Google Seattle Conference on Scalability (http://www.youtube.com/watch?gl=US&v=ZW5_eEKEC28)*, Seattle (Washington, E.U.A.), 2007.

[21] Joost, **www.joost.com**. *Online*, última visita em 28 de Julho de 2009.

[22] C. MacCarthaigh, **Joost Network Architecture**.*UK Network Operators' Forum Meeting*, Manchester (Inglaterra), 2007

[23] MTV, **www.mtv.com**. *Online*, última visita em 28 de Julho de 2009.

[24] A. Silberschatz, Henry F. Korth, S. Sudarshan, **Database System Concepts**. 5ª Edição, McGrawHill, p. 201 – 261, 2006

[25] Adobe – Adobe AIR, **http://get.adobe.com/air/**. *Online*, última visita em 28 de Julho de 2009.

[26] Adobe – Adobe Flex: Download Adobe Flex SDK, **http://www.adobe.com/cfusion/entitlement/index.cfm?e=flex3sdk**. *Online*, última visita em 28 de Julho de 2009.

[27] Adobe – Adobe AIR: Download Adobe AIR SDK, **http://www.adobe.com/cfusion/entitlement/index.cfm?e=airsdk**. *Online*, última visita em 28 de Julho de 2009.